
ABC of DevOps

— Key Practices and Methodologies —

About Me

- 22 years of software industry experience wearing multiple hats in Dev, QA and Management and Leadership
- DevOps Practitioner specializing in Azure and AWS DevOps technologies
- Avid Educator with a passion for imparting knowledge in the areas of:
 - DevOps
 - Network Automation
 - Cloud Computing
 - Automation Testing etc.

Let's start with the usual...

What is DevOps?

To quote Microsoft:

A portmanteau coined by combining development (Dev) and operations (Ops), DevOps is the union of people, process, and technology to continually provide value to customers.

Let's start with the usual...

Key points:

What?

- People
- Processes
- Technology

Why?

- To provide value to customers

Let's start with the usual...

- Value to the customers would be in terms of:
 - Faster delivery
 - Better quality
 - Better efficiency
 - Better security
 - Overall customer satisfaction
- Hence, can it be safely thought of as:
 - Evolution of agile practices?
 - As a missing piece of agile?
- It's an effort to take the innovations of the agile approach and apply them to operations processes

So who are the people involved?

And what is the challenge faced currently?

- Developers
 - Too busy worrying about meeting the deadlines
 - Finishing current release and moving onto the next release
 - Build something and throw it over the wall to the Operations team
- Operations (IT Teams)
 - Too busy worrying about stability and reliability of the build
 - Even at the cost of release delays
- This leads to the famous ***Wall of Confusion***

The Great Wall of Ch... err... Confusion...



DEVELOPMENT: SPEED

- Solve customer problems
- Innovate new products
- Deploy frequently
- Roll out changes fast

WALL OF CONFUSION



OPERATIONS: STABILITY

- Maximise uptime
- Minimise failures
- Meet compliance obligations
- Maintain security

What's the way out then?

The organizations have to follow the three pillared approach

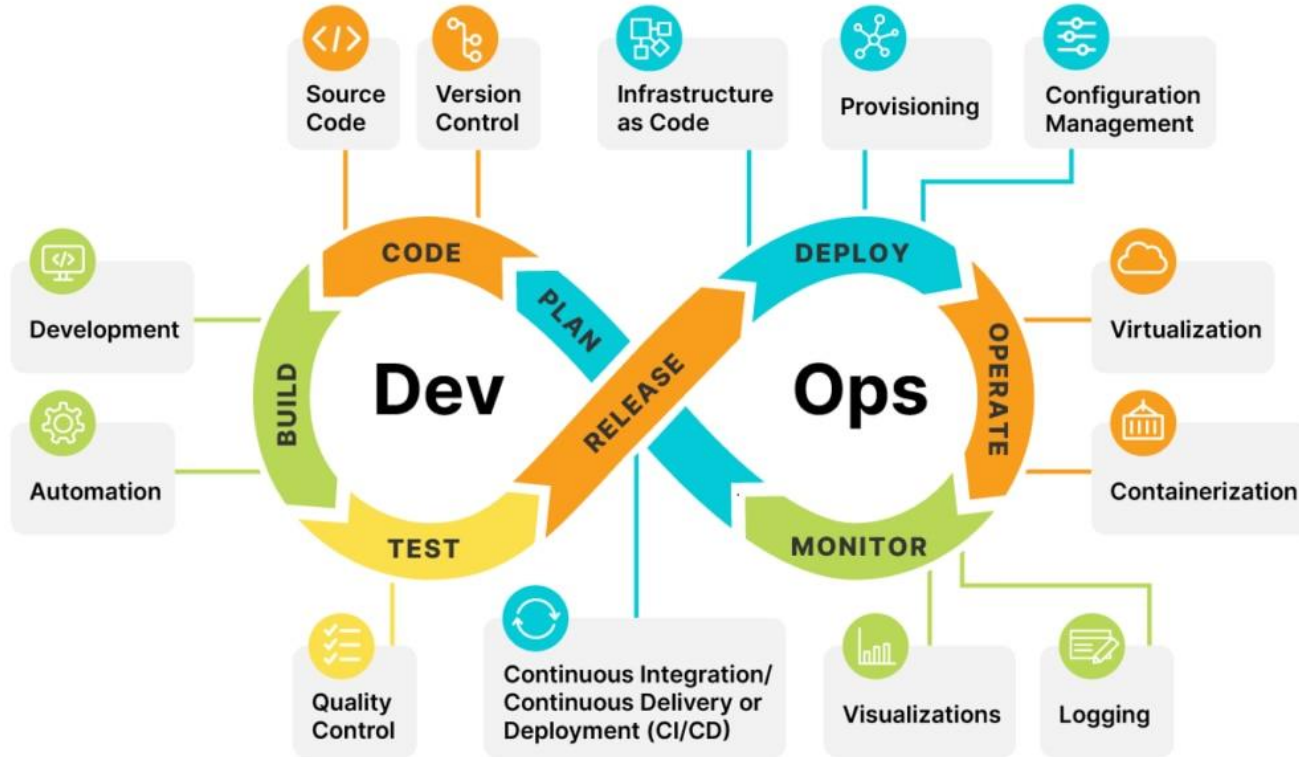
- Culture
 - Adopt the culture at the organization level to do what is required to reap the benefits DevOps
 - Enable people, motivate them and trust them to get the job done
- Technology
 - Adopt automation principles which is a key factor for DevOps success
 - Have an Automation first outlook
 - Automate, Automate and Automate and then Automate some more
 - There is no dearth of tools out there now a days
 - But choosing the right tool for the right job still remains a challenge

What's the way out then?

The organizations have to follow the three pillared approach

- Processes
 - Do not let the processes get in the way of experimentation and innovation
 - Streamline approvals, remove unwanted delays and enable customer feedback loops
- In essence
 - Break the silos
 - Bring people and teams together with a single objective of value to the customer in mind
 - Realign the processes to work towards this objective
 - Adapt the tools required to achieve this objective

And the wall is broken...



Continuous is the key...

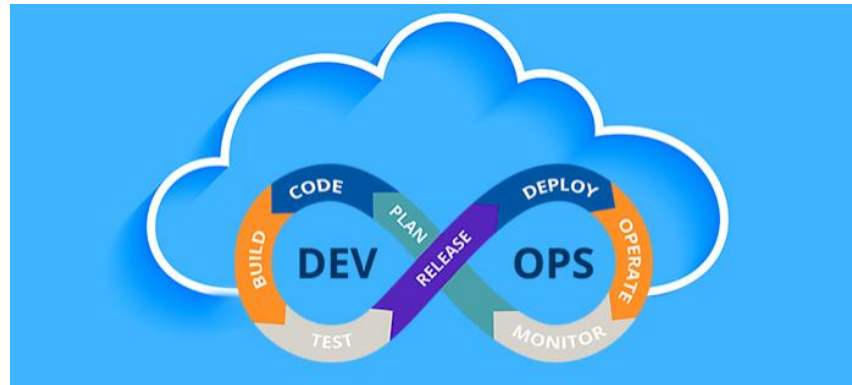
- When we think of DevOps, what comes to our mind if CI/CD
- Which stands for Continuous Integration/ Continuous Delivery or Deployment
- But why stop there?
- Why not?
 - Continuous Testing (Hidden somewhere in between CI and CD)
 - Continuous Provisioning
 - Continuous Configuration
 - Continuous Containerization
 - Continuous Orchestration
 - Continuous Monitoring

Where does Cloud fit in?

- If DevOps was cake, Cloud is Cherry on the cake
- Cloud makes DevOps even better
- DevOps processes can be very agile when implemented correctly
- But the limitations of an on-premise environment can take away all the advantage
- Imagine if an organization needs start a new software project or scale up a production application
- Nobody will have the infrastructure required for this just sitting there and waiting to be utilized
- It needs to be procured, installed and set up, causing unwanted delays

Where does Cloud fit in?

- Cloud infrastructure with its dynamic provisioning and scaling capabilities makes this process smooth
- While giving the advantage of pay as you go
- The cloud minimizes latency and enables centralized management via a unified platform
- A cloud platform allows DevOps teams to adapt to changing requirements



DevOps Engineer Journey - Shell Scripting

- Linux Shell Scripting
 - Shell scripting is an essential skill for DevOps Engineers...
 - To be able to automation routine repetitive tasks
 - To write custom scripts that enhance the power of existing tools
 - To implement version control best practices by writing git hooks

```
#!/bin/sh
#
# An example hook script to verify what is about to be committed.
# Called by "git commit" with no arguments. The hook should
# exit with non-zero status after issuing an appropriate message if
# it wants to stop the commit.
#
# To enable this hook, rename this file to "pre-commit".

if git rev-parse --verify HEAD >/dev/null 2>&1
then
    against=HEAD
else
    # Initial commit: diff against an empty tree object
    against=$(git hash-object -t tree /dev/null)
fi

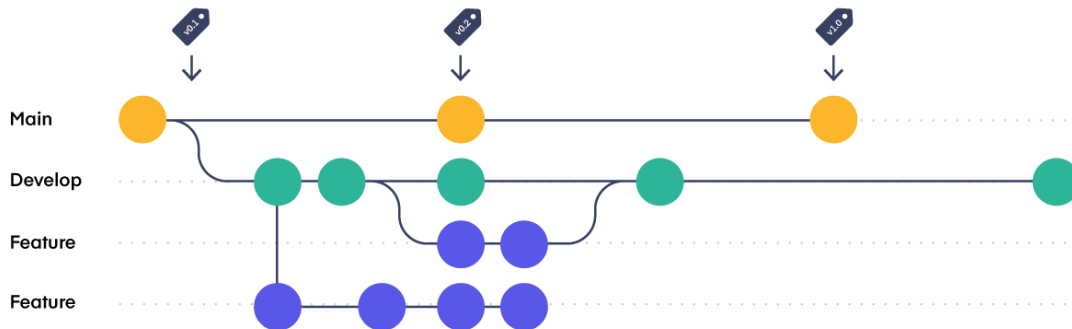
# If you want to allow non-ASCII filenames set this variable to true.
allownonascii=$(git config --type=bool hooks.allownonascii)

# Redirect output to stderr.
exec 1>&2
```

DevOps Engineer Journey - Git

- Software Version Control

- Git is a distributed version control system widely used in organizations
- To be able to help the developers set up repositories
- Create and manage branches
- Handle releases from the repositories



DevOps Engineer Journey - Cloud Alternatives

- Azure Repos
- AWS CodeCommit

DevOps Engineer Journey - CI/CD

- CI/CD
 - Continuous Integration is the process of multiple developers attempting to merge their code changes with the main code repository of a project
 - Continuous delivery is the next extension of continuous integration
 - The delivery phase is responsible for packaging an artifact together to be delivered to end-users
- Some popular CI/CD tools are
 - Jenkins
 - CircleCI
 - Travis CI
 - Azure Pipelines
 - AWS Pipelines
 - Github Actions/ GitLab CI CD Pipeline

DevOps Engineer Journey - Jenkins

- Jenkins
 - Jenkins is an open source automation server
 - It helps automate the parts of software development related to building, testing, and deploying
 - This facilitates continuous integration, and continuous delivery
- Jenkins Build Jobs
 - A Jenkins build job contains the configuration for automating a specific task or step in the application building process
 - These tasks include gathering dependencies, compiling, archiving, or transforming code, and testing and deploying code in different environments
 - These jobs can be created either as freestyle projects or pipelines

DevOps Engineer Journey - Jenkins Pipeline

Pipeline

Definition

Pipeline script

Script

```
1 pipeline {
2     agent any
3     stages {
4         stage('Build') {
5             steps {
6                 // Get some code from a GitHub repository
7                 git url: 'https://github.com/naiveskill/devops.git', branch: 'main'
8                 // Change file permission
9                 sh "chmod +x -R ./jenkins"
10                // Run shell script
11                sh "./jenkins/script/scripted_pipeline_ex_2.sh"
12            }
13        }
14    }
15 }
```

DevOps Engineer Journey - Cloud Alternatives

- Azure Pipelines
- AWS CodePipeline

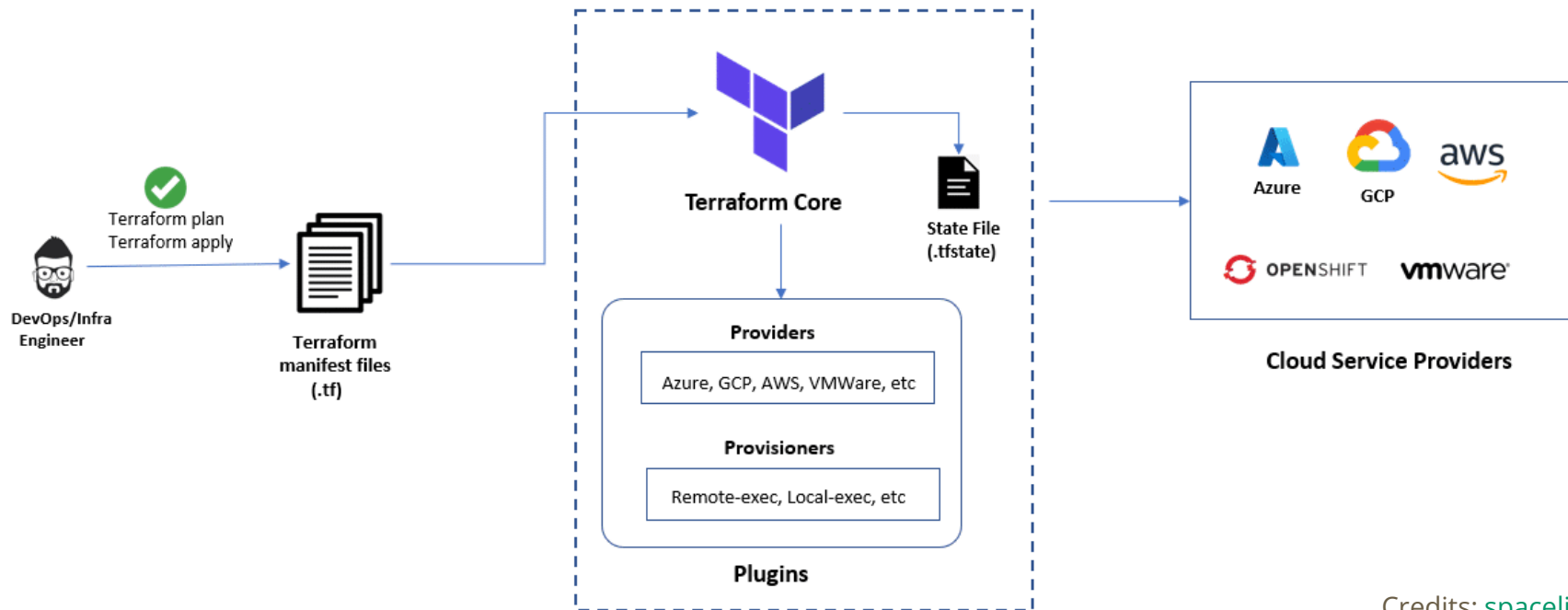
DevOps Engineer Journey - IaC

- Continuous Provisioning

- Infrastructure as code (IaC) is a key success factor behind continuous provisioning
- IaC is the process of managing and provisioning computer data center resources through machine-readable definition files
- IaC makes much more sense in the cloud than on-premise
- Some IaC tools are:
 - AWS CloudFormation
 - Google Cloud Deployment Manager
 - HashiCorp Terraform
 - Microsoft Azure Resource Manager

DevOps Engineer Journey

Terraform Architecture



Credits: [spacelift](#)

DevOps Engineer Journey - Terraform

- Terraform is a popular IaC tool
 - Users define and provide data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language, or optionally JSON
- Providers
 - A provider in Terraform is a plugin that enables interaction with an API
- Registry
 - The Terraform Registry is an interactive resource for discovering a wide selection of integrations (providers), configuration packages (modules)

```
main.tf > ...
provider "aws" {
    region = "us-east-1"
}

0 references
resource "aws_instance" "ec2" {
    ami = "ami-02354e95b39ca8dec"
    instance_type = "t2.micro"
}
```

DevOps Engineer Journey - Cloud Alternatives

- ARM Templates
- AWS CloudFormation

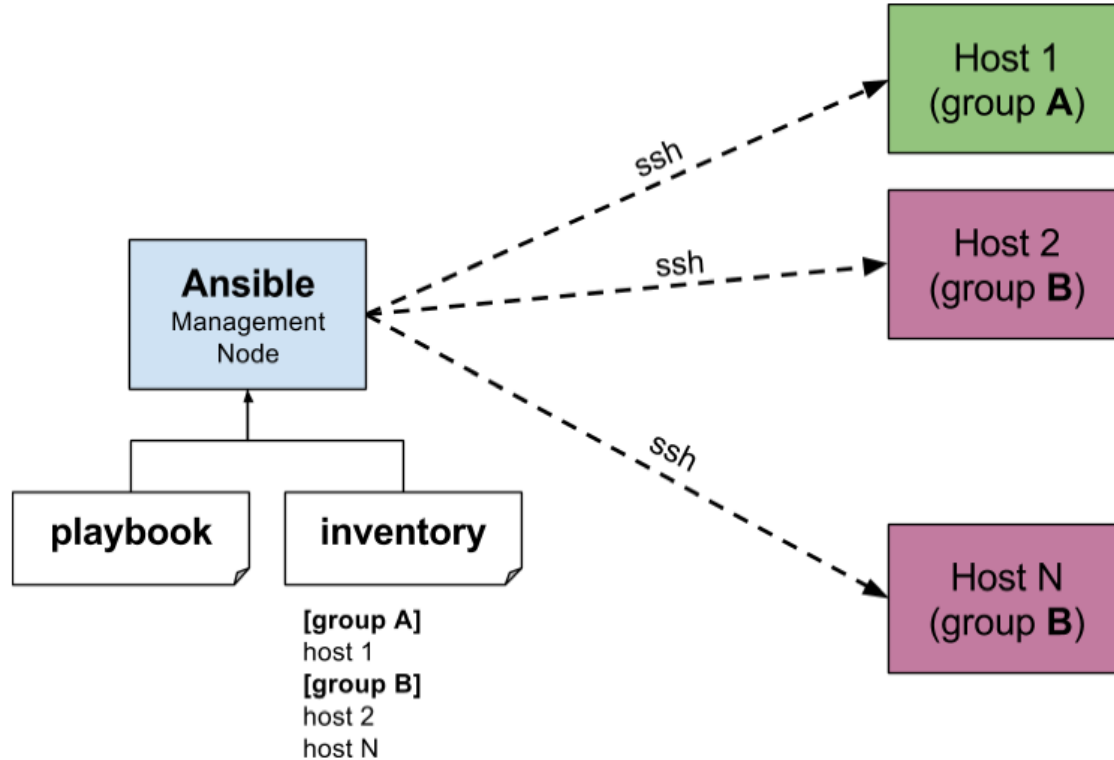
DevOps Engineer Journey - Config Management

- Continuous Configuration Management
 - Process of maintaining systems, such as computer hardware and software, in a desired state
 - Method of ensuring that systems perform in a manner consistent with expectations over time
- Configuration Management Tools
 - Enable changes and deployments to be faster, repeatable, scalable, predictable
 - Able to maintain the desired state, which brings controlled assets into an expected state
 - This is also one of the IaC principles, where the desired state is specified in code
- Some well known tools include
 - Ansible
 - Chef
 - Puppet
 - Salt

DevOps Engineer Journey - Ansible

- CM tools generally work with a central node pushing the config changes to remote machines which need to be put in a desired state
- These tools can be classified as agent-based or agentless
- Agent-based tools require an agent to be installed on the target machines
- With agentless, you can centrally deploy and configure applications
- Ansible is an agent less tool written in Python
- Makes use of small python programs called modules to carry out various tasks on remote machines

DevOps Engineer Journey - Ansible Architecture



DevOps Engineer Journey - Ansible Playbook

```
- hosts: webservers
  sudo: yes

  vars:
    app_name: PleaseDeployMe
    repo_url: https://github.com/username/repo_name.git
    repo_remote: origin
    repo_version: master
    webapps_dir: /deployed
    virtualenv_root: /deployed/PleaseDeployMe/mac
  tasks:

    - name: git pull project
      git: repo={{repo_url}} dest={{webapps_dir}}/{{app_name}} version=master

      notify:
        - restart app

    - name: install things
      pip: name=virtualenv

    - name: create virtualenv
      command: virtualenv /deployed/PleaseDeployMe/venv

    - name: activate virtualenv
      command: /bin/bash /deployed/PleaseDeployMe/venv/bin/activate

    - pip: requirements=/deployed/{{app_name}}/requirements.txt virtualenv=/deployed/{{app_name}}/mac

    - name: run supervisord
      command: "supervisord -c /deployed/PleaseDeployMe/supervisord.conf"

    - name: begin flask app
      supervisorctl: name=flask_app state=started
```

DevOps Engineer Journey - Cloud Alternatives

- Azure Automation
- AWS CloudFormation

DevOps Engineer Journey - YAML

- YAML is a human-readable data serialization language
- It is commonly used for configuration files and in applications where data is being stored or transmitted
- It uses Python-style indentation to indicate nesting
- It does not require quotes around most string values
- You can think of it as further simplified version of JSON with the curly braces and quotes removed

DevOps Engineer Journey - YAML

YAML

```
apis:
  - name: login
    port: 8080
  - name: profile
    port: 8090
```

XML

```
<apis>
  <api>
    <name>login</name>
    <port>8080</port>
  </api>
  <api>
    <name>profile</name>
    <port>8090</port>
  </api>
</apis>
```

JSON

```
{
  "apis": [
    {
      "name": "login",
      "port": 8080
    },
    {
      "name": "profile",
      "port": 8090
    }
  ]
}
```

DevOps Engineer Journey - Ansible Command Module

```
(ansible) expert@devnet-lab:~$ ansible-doc command  
> COMMAND (/home/expert/venvs/ansible/lib/python3.9/site-packages/ansible/modules/commands/command.py)
```

The 'command' module takes the command name followed by a list of space-delimited arguments. The given command will be executed on all selected nodes. The command(s) will not be processed through the shell, so variables like '\$HOME' and operations like '<', '>', '|', ';' and '&' will not work. Use the [shell] module if you need these features. To create 'command' tasks that are easier to read than the ones using space-delimited arguments, pass parameters using the 'args' L(task keyword, ../reference_appendices/playbooks_keywords.html#task) or use 'cmd' parameter. Either a free form command or 'cmd' parameter is required, see the examples. For Windows targets, use the [win_command] module instead.

- * This module is maintained by The Ansible Core Team
- * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- argv

Passes the command as a list rather than a string.
Use 'argv' to avoid quoting values that would otherwise be interpreted incorrectly (for example "user name"). Only the string or the list form can be provided, not both. One or the other must be provided.
[Default: (null)]
type: list
version_added: 2.6

DevOps Engineer Journey - Command Module

```
GNU nano 4.8 /home/expert/venvs/ansible/lib/python3.9/site-packages/ansible/modules/commands/command.py

import datetime
import glob
import os
import shlex

from ansible.module_utils.basic import AnsibleModule
from ansible.module_utils._text import to_native, to_bytes, to_text
from ansible.module_utils.common.collections import is_iterable

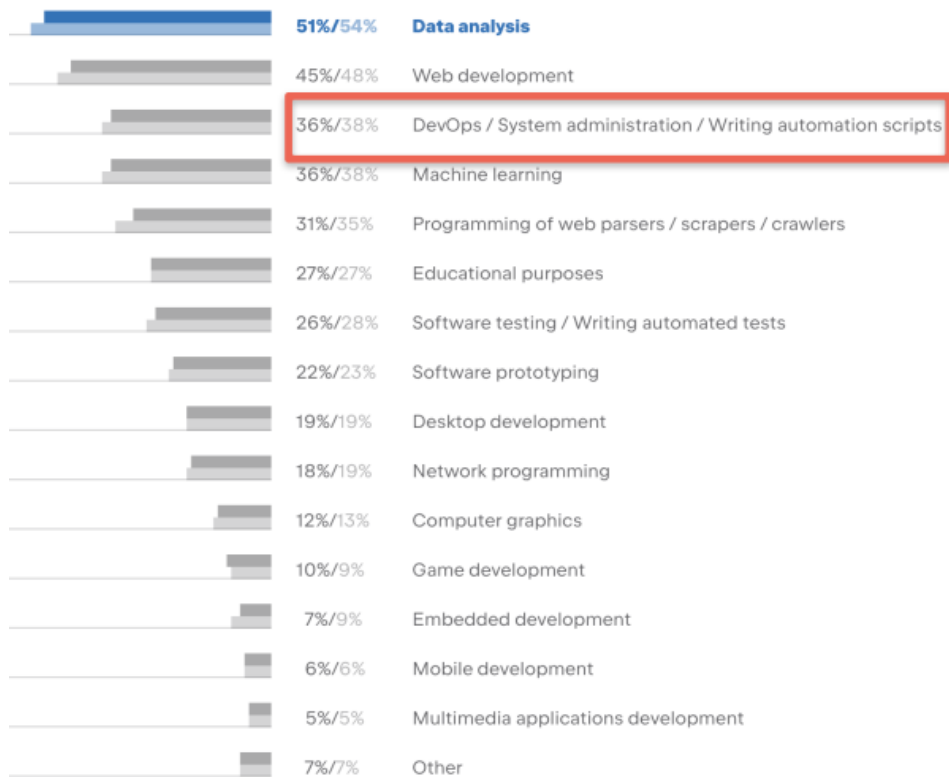
def check_command(module, commandline):
    arguments = {'chown': 'owner', 'chmod': 'mode', 'chgrp': 'group',
                 'ln': 'state=link', 'mkdir': 'state=directory',
                 'rmdir': 'state=absent', 'rm': 'state=absent', 'touch': 'state=touch'}
    commands = {'curl': 'get_url or uri', 'wget': 'get_url or uri',
                'svn': 'subversion', 'service': 'service',
                'mount': 'mount', 'rpm': 'yum, dnf or zypper', 'yum': 'yum', 'apt-get': 'apt',
                'tar': 'unarchive', 'unzip': 'unarchive', 'sed': 'replace, lineinfile or template',
                'dnf': 'dnf', 'zypper': 'zypper'}
    become = ['sudo', 'su', 'pbrun', 'pfexec', 'runas', 'pmrun', 'machinectl']
    if isinstance(commandline, list):
        command = commandline[0]
    else:
        command = commandline.split()[0]
    command = os.path.basename(command)

    disable_suffix = "If you need to use command because {mod} is insufficient you can add" \
                     " 'warn: false' to this command task or set 'command_warnings=False' in" \
                     " ansible.cfg to get rid of this message."
    substitutions = {'mod': None, 'cmd': command}
```

DevOps Engineer Journey - Python

- Programming is becoming an essential skill for DevOps engineers
- It doesn't mean DevOps engineers have to develop full-fledged applications like developers
- Python is one of the best programming languages for Automation in DevOps
- DevOps teams use Python for automating repetitive tasks:
 - Infrastructure provisioning
 - API-driven deployments like writing new or extending existing Ansible modules
 - Coding CI/CD workflows and much more

DevOps Engineer Journey - Python

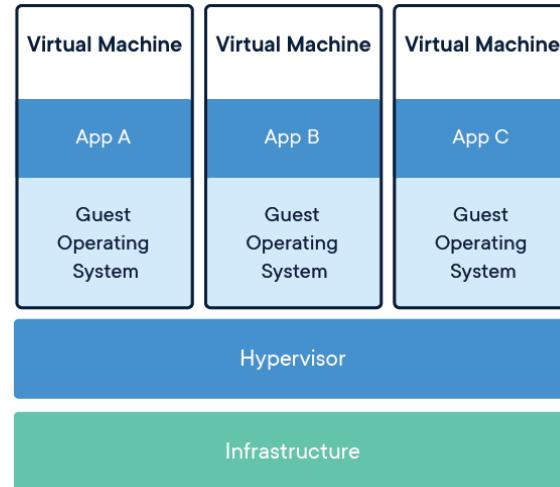
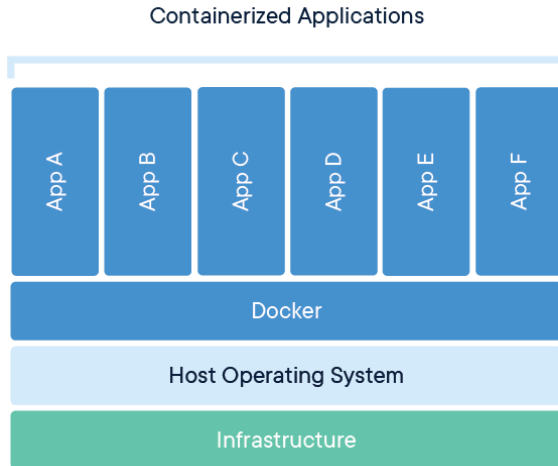


DevOps Engineer Journey - Python

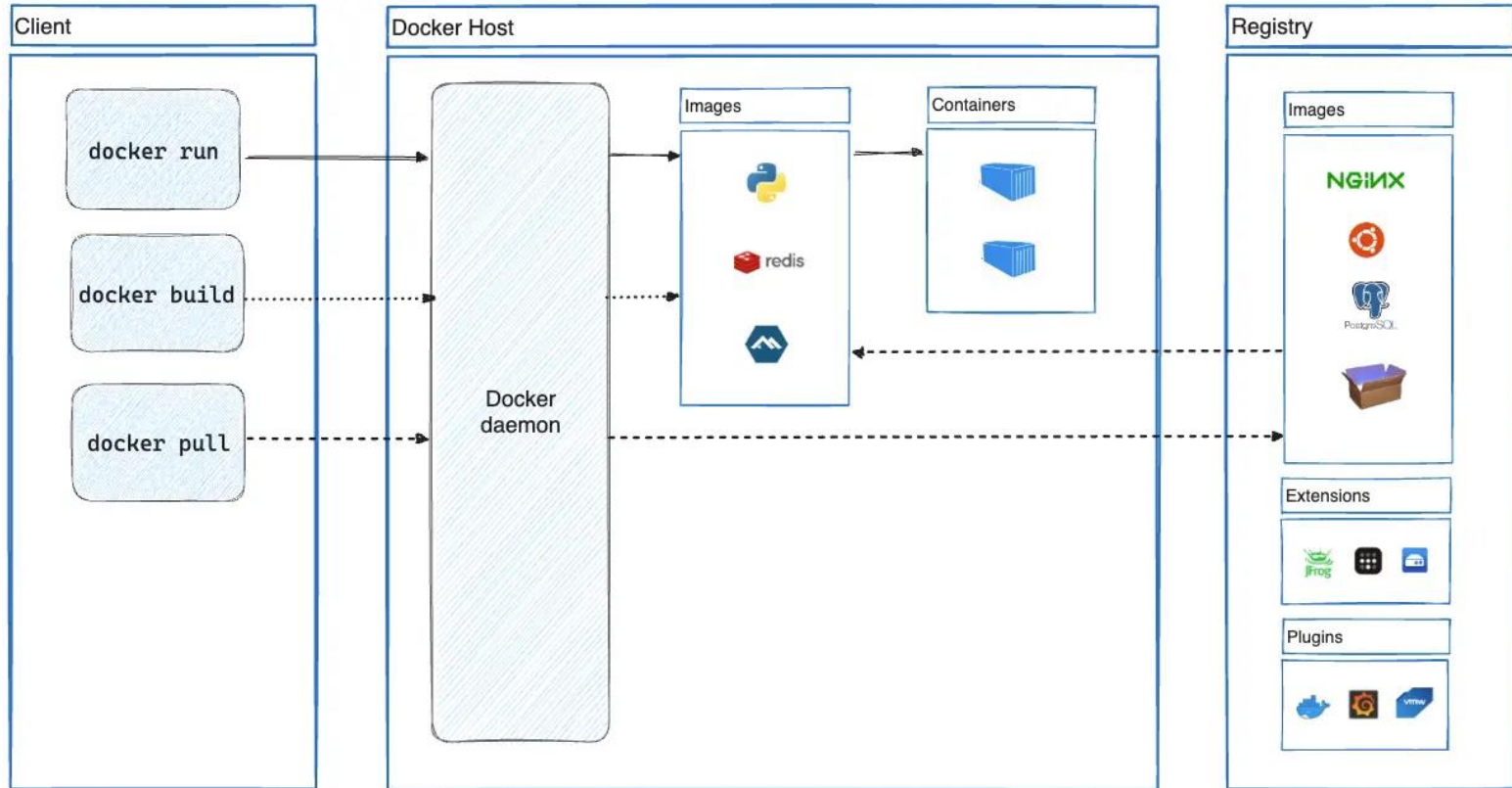
- Important Python Libraries used in DevOps
 - os
 - platform
 - subprocess
 - sys
 - psutil
 - re (Regular Expression)
 - Requests and urllib3
 - logging
 - getpass
 - paramiko
 - JSON
 - PyYAML
 - pandas
 - smtplib

DevOps Engineer Journey - Docker

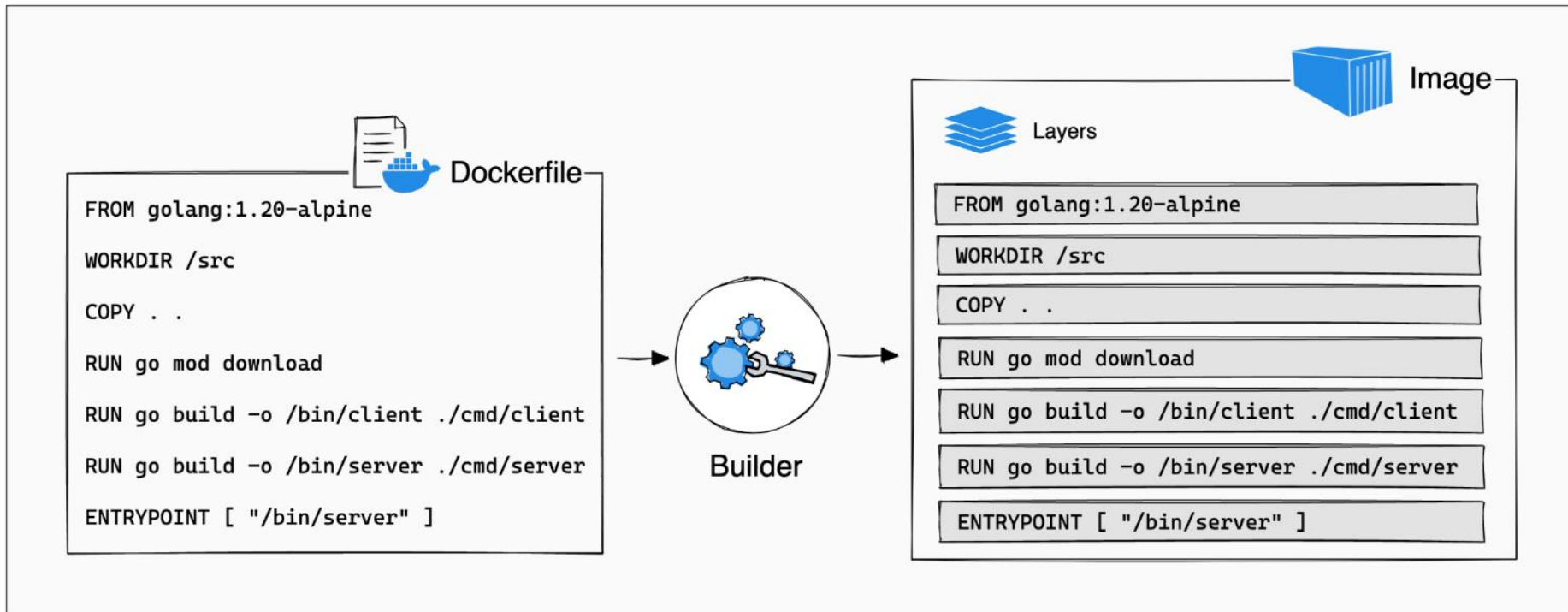
- Continuous Containerization
 - Containerization is a type of virtualization
 - In this all the components of an application are bundled into a single container image
 - This image can be run in isolated user space on the same shared operating system
 - Docker is a platform designed to help developers build, share, and run container applications



DevOps Engineer Journey - Docker Architecture



DevOps Engineer Journey - Dockerizing a go app



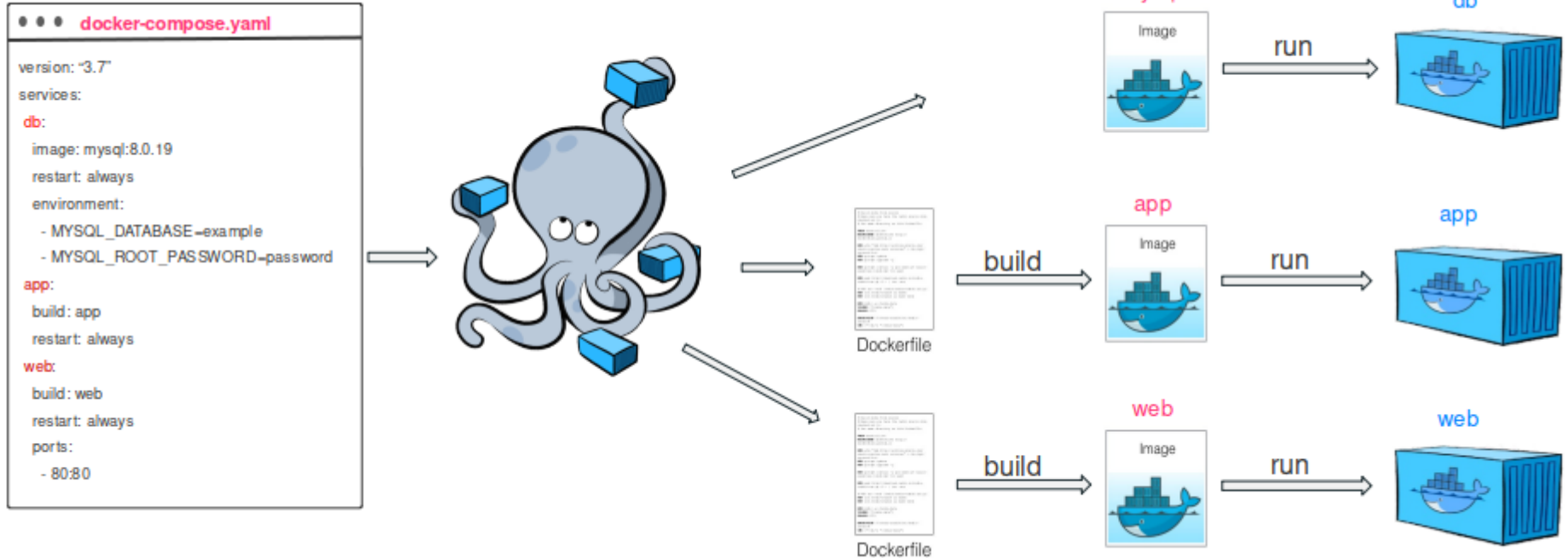
DevOps Engineer Journey - Docker Compose

- Docker Compose is a tool for defining and running multi-container applications
- Compose simplifies the control of your entire application stack
- Makes it easy to manage services, networks, and volumes in a single, comprehensible YAML configuration file
- With a single command, we can create and start all the services from your configuration file

DevOps Engineer Journey - Docker Compose

- Compose works in all environments like production, staging, development, testing, as well as CI workflows
- It also has commands for managing the whole lifecycle of your application:
 - Start, stop, and rebuild services
 - View the status of running services
 - Stream the log output of running services
 - Run a one-off command on a service

DevOps Engineer Journey - Docker Compose



DevOps Engineer Journey - WordPress with Compose

```
1  version: '3.3'
2
3  services:
4    db:
5      image: mysql:5.7
6      volumes:
7        - db_data:C:\Users\User\Desktop\dcompose
8      restart: always
9      environment:
10        MYSQL_ROOT_PASSWORD: rootwordpress
11        MYSQL_DATABASE: wordpress
12        MYSQL_USER: wordpress
13        MYSQL_PASSWORD: wordpress
14
15    wordpress:
16      depends_on:
17        - db
18      image: wordpress:latest
19      ports:
20        - "8000:80"
21      restart: always
22      environment:
23        WORDPRESS_DB_HOST: db:3306
24        WORDPRESS_DB_USER: wordpress
25        WORDPRESS_DB_PASSWORD: wordpress
26  volumes:
27    db_data:
```

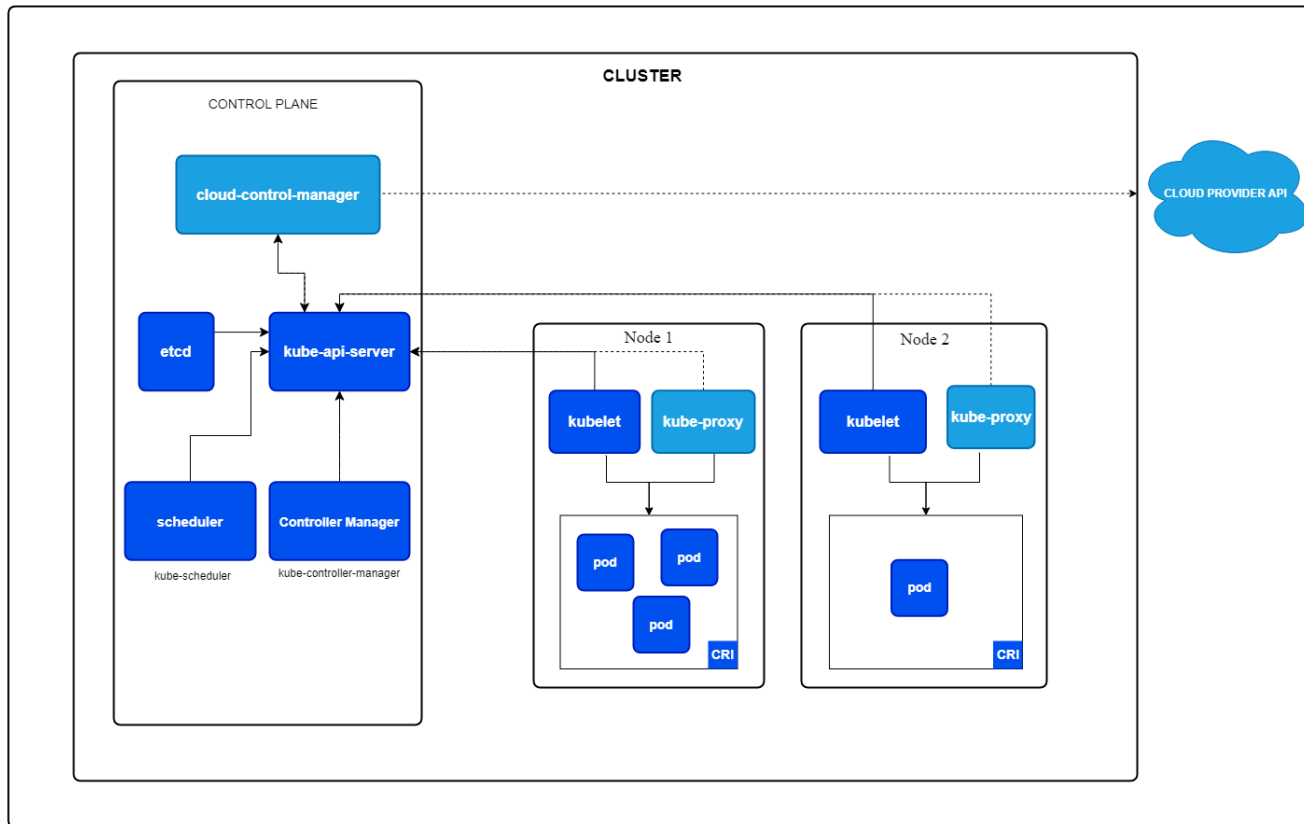
DevOps Engineer Journey - Cloud Alternatives

- Azure Container Registry
- Amazon Elastic Container Registry (Amazon ECR) - AWS

DevOps Engineer Journey - Orchestration

- Container orchestration is a part of a CI/CD pipeline in DevOps
- It is used to automate the management, deployment, scaling and networking of containers
- Kubernetes is the most popular and widely used container orchestration platform
- Main concept behind Kubernetes is Cluster
- Cluster is collection of Nodes (VMs)
- Each node consist of Pods which host containers

DevOps Engineer Journey - Cluster Architecture

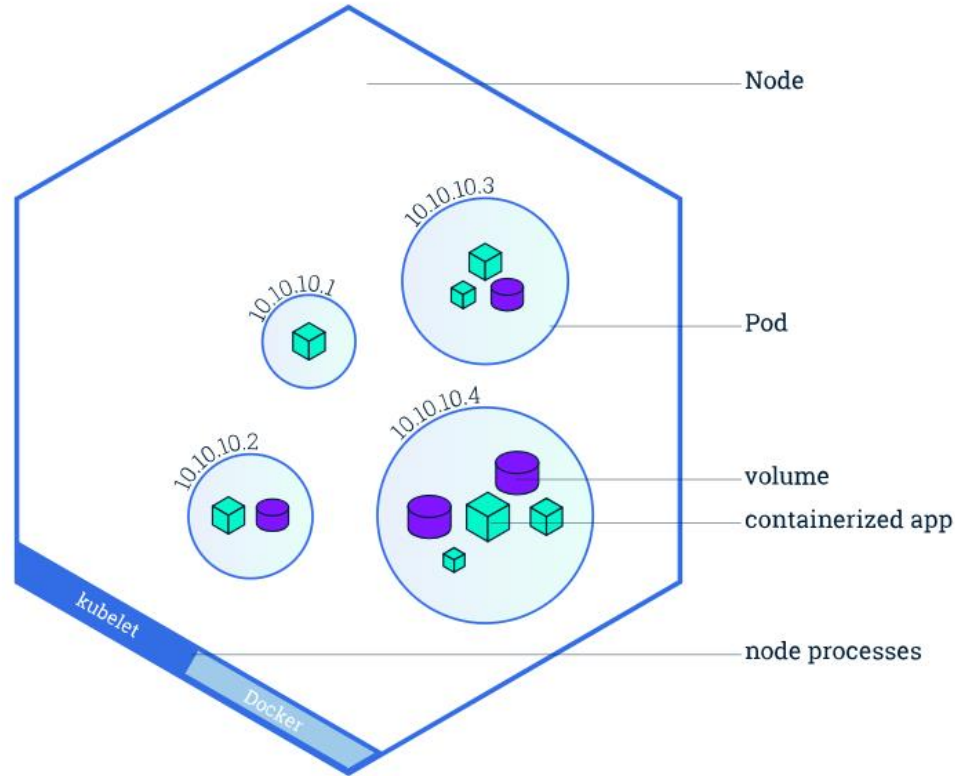


DevOps Engineer Journey - Cluster Architecture



Credits: [snexplores](#)

DevOps Engineer Journey - Cluster Architecture



DevOps Engineer Journey - Kubernetes Deployment

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 80
```

DevOps Engineer Journey - Kubernetes Service

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app.kubernetes.io/name: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

DevOps Engineer Journey - Cloud Alternatives

- Azure Kubernetes Service (AKS)
- Amazon Elastic Kubernetes Service
- Red Hat OpenShift enterprise Kubernetes container platform

DevOps Engineer Journey - Monitoring

- Continuous Monitoring in DevOps helps organizations track the operational performance of the applications
- Provides immediate feedback and insight into performance issues with the applications across the geographies
- Popular monitoring tools used in DevOps are:
 - Nagios
 - Offers monitoring and alerting services for servers, switches, applications and services
 - It alerts users when things go wrong and alerts them a second time when the problem has been resolved
 - Prometheus
 - Provides monitoring and alerting functionality for cloud-native environments, including Kubernetes

DevOps Engineer Journey - Nagios

Nagios®

General

- Home
- Documentation

Monitoring

- Tactical Overview
- Service Detail
- Host Detail
- Status Overview
- Status Summary
- Status Grid
- Status Map
- 3-D Status Map
- Service Problems
- Host Problems
- Network Outages
- Comments
- Downtime
- Process Info
- Performance Info
- Scheduling Queue

Reporting

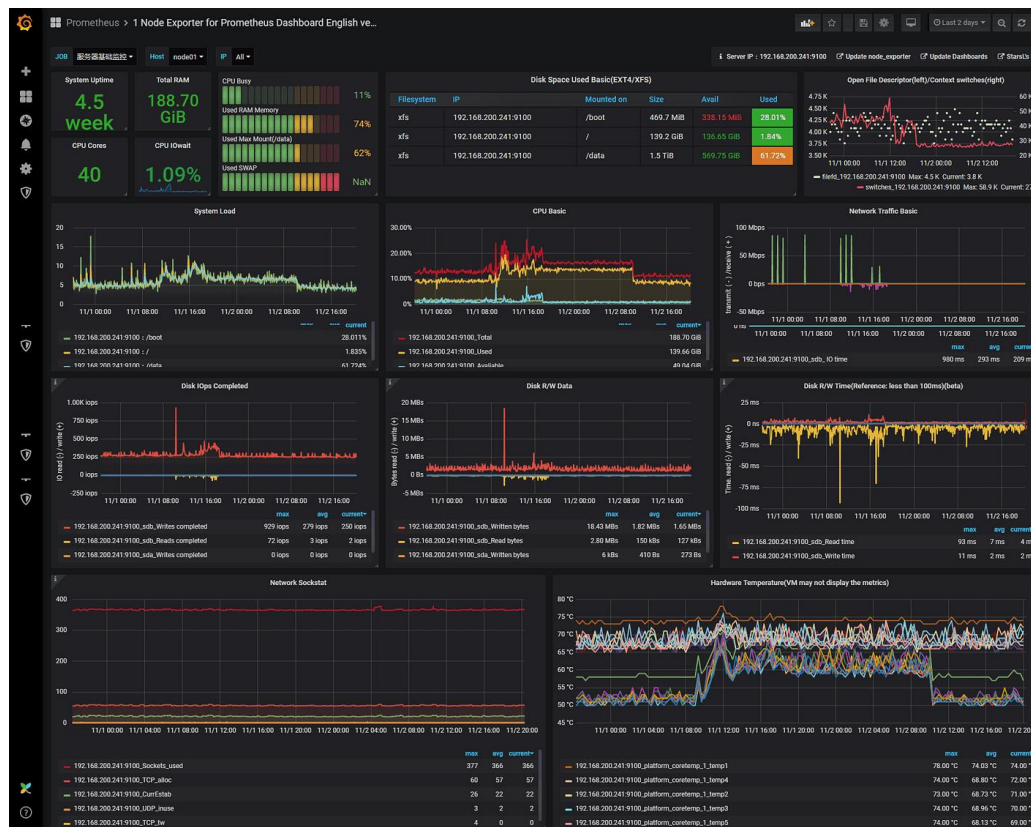
- Trends
- Availability
- Alert Histogram
- Alert History
- Alert Summary
- Notifications
- Event Log

Configuration

- View Config

Monitor						
webprod03	Check Users	OK	01-26-2007 14:58:59	0d 4h 53m 23s	1/4	USERS OK - 1 users currently logged in
	Current Load	OK	01-26-2007 14:59:54	0d 4h 53m 23s	1/4	OK - load average: 0.21, 0.08, 0.05
	Memory Usage	OK	01-26-2007 14:55:29	0d 4h 53m 23s	1/4	OK: Memory Usage 56% - Total: 511 MB, Used: 287 MB, Free: 224 MB
	PING	OK	01-26-2007 14:56:14	0d 4h 50m 23s	1/4	PING OK - Packet loss = 0%, RTA = 0.16 ms
	Root Partition	OK	01-26-2007 14:57:09	0d 4h 50m 33s	1/4	DISK OK [243816 kB (5%) free on /dev/sda2]
	SWAP Usage	OK	01-26-2007 14:57:44	0d 4h 50m 33s	1/4	Swap ok - (null) 0% (0 out of 16386)
webprod04	Total Processes	OK	01-26-2007 14:58:29	0d 4h 50m 33s	1/4	OK - 95 processes running
	Xen Virtual Machine Monitor	CRITICAL	01-26-2007 14:59:04	0d 0h 44m 34s	4/4	Critical Xen VMs Usage - Total NB: 0 - detected VMs:
	Check Users	OK	01-26-2007 14:59:54	0d 0h 15m 33s	1/4	USERS OK - 2 users currently logged in
	Current Load	OK	01-26-2007 14:55:34	0d 0h 14m 53s	1/4	OK - load average: 0.30, 0.60, 0.44
	Memory Usage	OK	01-26-2007 14:56:19	0d 0h 14m 13s	1/4	OK: Memory Usage 37% - Total: 511 MB, Used: 190 MB, Free: 321 MB
	PING	OK	01-26-2007 14:57:10	0d 0h 13m 23s	1/4	PING OK - Packet loss = 0%, RTA = 0.27 ms
webprod05	Root Partition	OK	01-26-2007 14:57:49	0d 0h 12m 43s	1/4	DISK OK [3948940 kB (94%) free on /dev/sda2]
	SWAP Usage	OK	01-26-2007 14:58:34	0d 0h 11m 53s	1/4	Swap ok - (null) 0% (0 out of 16386)
	Total Processes	OK	01-26-2007 14:59:09	0d 0h 16m 22s	1/4	OK - 250 processes running
	Xen Virtual Machine Monitor	WARNING	01-26-2007 14:58:54	0d 0h 1m 33s	4/4	Warning Xen VMs Usage - Total NB: 1 - detected VMs: migrating-xen-vm4
	PING	OK	01-26-2007 14:55:39	0d 0h 24m 58s	1/4	PING OK - Packet loss = 0%, RTA = 0.25 ms
	Xen Virtual Machine Monitor	OK	01-26-2007 14:59:54	0d 0h 0m 33s	1/4	OK: Xen Hypervisor "webprod05" is running 4 Xen VMs: xen-vm1 xen-vm2 xen-vm3 xen-vm4
xen-vm1	Check Users	OK	01-26-2007 14:58:09	0d 0h 17m 23s	1/4	USERS OK - 1 users currently logged in
	Current Load	OK	01-26-2007 14:57:54	0d 3h 16m 21s	1/4	OK - load average: 1.54, 1.09, 0.48
	Memory Usage	OK	01-26-2007 14:58:39	0d 3h 15m 41s	1/4	OK: Memory Usage 8% - Total: 8195 MB, Used: 676 MB, Free: 7519 MB
	PING	OK	01-26-2007 14:59:15	0d 3h 15m 21s	1/4	PING OK - Packet loss = 0%, RTA = 0.49 ms
	Root Partition	OK	01-26-2007 14:59:59	0d 3h 14m 51s	1/4	DISK OK [4196280 kB (99%) free on udev]
	SWAP Usage	OK	01-26-2007 14:55:44	0d 3h 14m 1s	1/4	Swap ok - (null) 0% (0 out of 2055)
xen-vm2	Total Processes	OK	01-26-2007 14:57:29	0d 0h 18m 3s	1/4	OK - 88 processes running
	Check Users	OK	01-26-2007 14:57:15	0d 3h 7m 41s	1/4	USERS OK - 0 users currently logged in
	Current Load	OK	01-26-2007 14:57:59	0d 3h 7m 1s	1/4	OK - load average: 0.00, 0.00, 0.00
	Memory Usage	OK	01-26-2007 14:58:44	0d 3h 6m 21s	1/4	OK: Memory Usage 6% - Total: 1023 MB, Used: 64 MB, Free: 958 MB
	PING	OK	01-26-2007 14:59:19	0d 0h 48m 14s	1/4	PING OK - Packet loss = 0%, RTA = 0.43 ms
	Root Partition	OK	01-26-2007 15:00:05	0d 1h 15m 4s	1/4	DISK OK [524220 kB (99%) free on udev]
	SWAP Usage	OK	01-26-2007 14:55:49	0d 3h 9m 41s	1/4	Swap ok - (null) 0% (0 out of 2055)
	Total Processes	OK	01-26-2007 14:56:34	0d 3h 9m 1s	1/4	OK - 52 processes running

DevOps Engineer Journey - Prometheus with Grafana



DevOps Engineer Journey - Cloud Alternatives

- Azure Monitor
- Amazon CloudWatch

What next?

- DevSecOps
 - The practice of integrating security testing at every stage of the software development process
 - It includes tools and processes that encourage collaboration between developers, security specialists, and operation teams to build software that is both efficient and secure
 - Focuses on reducing the risk of vulnerabilities in software by integrating security early in the development process

What next?

- NetDevOps

“Every time we implement a network change, something goes wrong.”

“The network is too critical to upgrade.”

“Joe is on vacation and without him, we cannot make changes to the network.”

What next?

- NetDevOps
 - Brings the culture, technical methods, strategies and best practices of DevOps to Networking
 - Orchestrates and automates network changes to reduce the network delivery lifecycle
 - Treats the network as code to allow for version control
 - Reliably tests changes to make sure of quality and stability

What next?

- MLOps
 - Stands for Machine Learning Operations
 - Is a core function of Machine Learning engineering
 - Focused on streamlining the process of taking machine learning models to production
 - Maintaining and monitoring them

What next?

- AIOps
 - Stands for Artificial Intelligence for IT Operations
 - Harnesses artificial intelligence and machine learning to elevate IT operations through automation of:
 - Routine tasks
 - Prediction and mitigation of incidents
 - Optimization of infrastructure and application performance
 - Proactively identifies and resolves issues, minimizing downtime and enhancing system reliability by analyzing extensive operational data

What next?

- ArchOps (Architecture + Operations)
 - Unites software architecture and IT operations
 - Guarantees that architectural choices align with operational demands
 - Promotes architectural excellence across the software development lifecycle
 - Seeks to harmonize architectural decisions with operational necessities, yielding scalable, maintainable, and efficient software systems

Industry Use Cases

Company	Experience
Netflix	Has 600+ services in production. Deploys 100 times per day.
Uber	Has 1,000+ services in production. Deploys several thousand times each week.
WeChat	Has 3,000+ services in production. Deploys 1,000 times a day.

Industry Use Cases

Netflix invested heavily in DevOps. Developers build, deploy and operate their own server clusters and are accountable when things go wrong.

Industry Use Cases

- Deployment at Netflix is completely automated
- When a service needs to be deployed, the developer first pushes the code to a source code repository
- The code push is picked up by Jenkins, which subsequently performs a build producing an application package
- Then, a fresh VM image (AMI) is produced based on a base image (containing a Linux distribution) and software that all Netflix servers run, including a JVM and Tomcat, possibly further customized by the team
- On top of this base install, the application package is installed
- From this, an AMI is produced and registered with the system

A - Z of DevOps

- A - Ansible
- B - Bitbucket
- C - Chef
- D - Docker
- E - Elasticsearch
- F - Finessse (Scanning tool)
- G - Grafana
- H - HCL (Hashicorp Configuration Language)
- I - Infrastructure as Code
- J - Jenkins
- K - Kubernetes
- L - Logstash
- M - Microservices

A - Z of DevOps

- N - Nagios
- O - Open Source
- P - Prometheus
- Q - Quality Assurance
- R - Raygun (Application Monitoring)
- S - SignalFx (Monitoring, Alerting)
- T - Terraform
- U - Urbancode (IBM's Deployment Automation Tool)
- V - Virtualization
- W - Webpack (JS Module Bundler)
- X - XML
- Y - YAML
- Z - ZAP (Zed Attack Proxy)

Thank you