DEVOPS

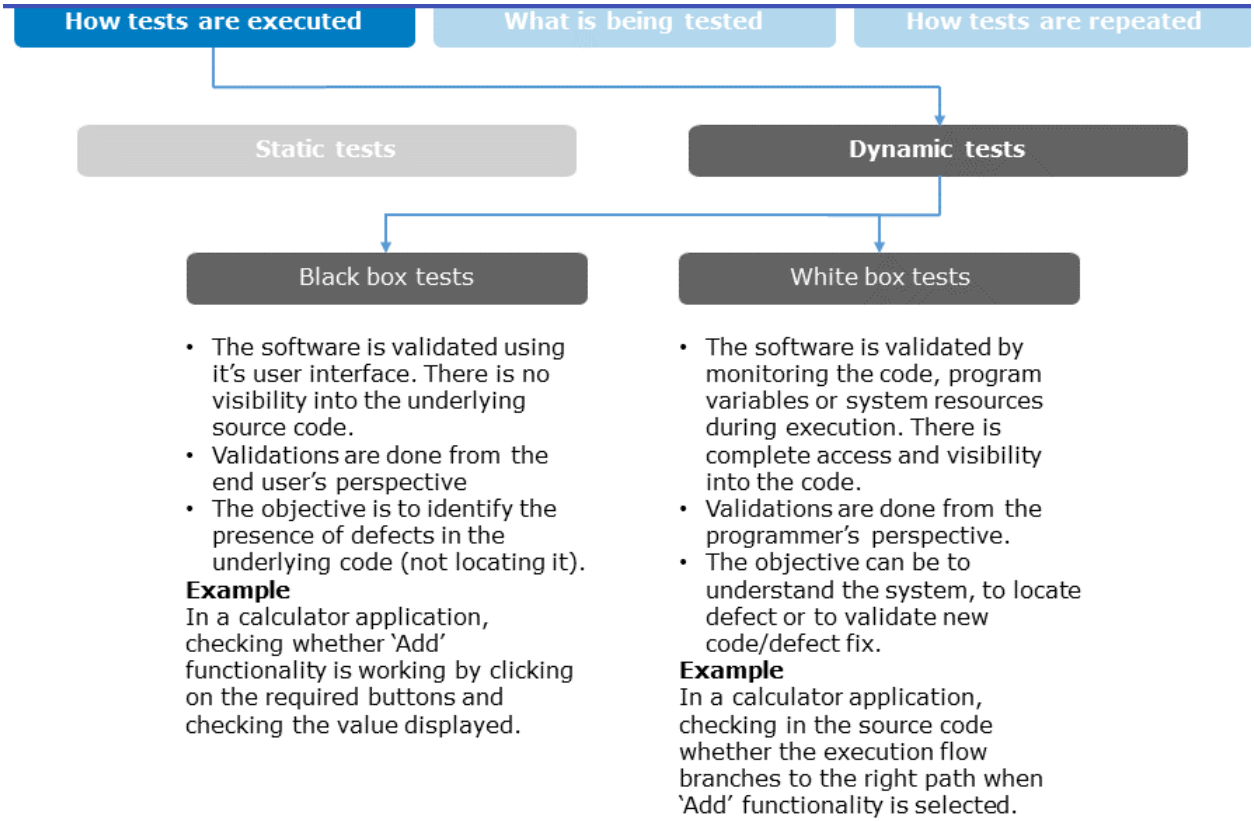
Contents

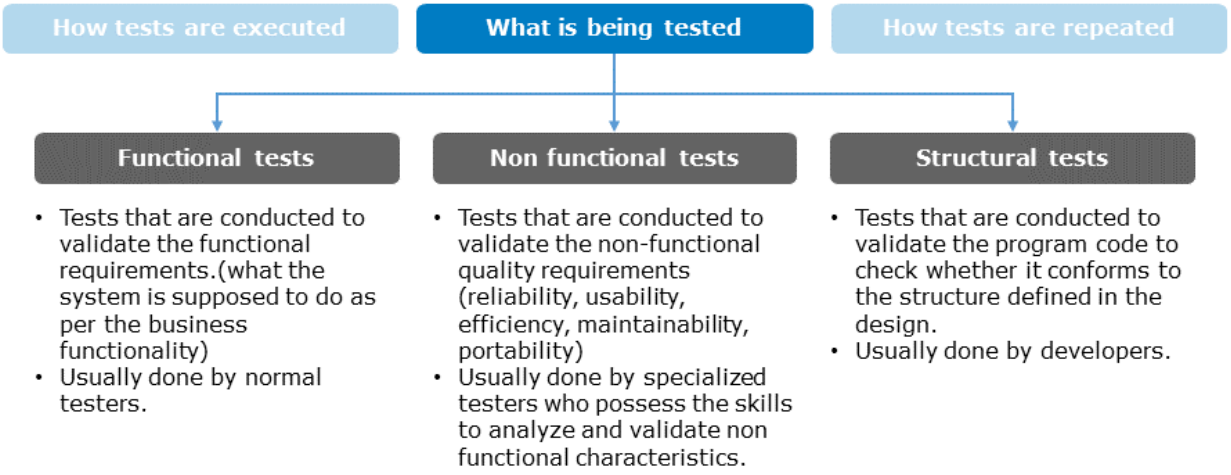
Software Testing

Software testing is a set of processes and tasks that take place throughout the software development life cycle. It helps to reduce the risk of failures that may occur during operational use and, thus, ensure the quality of the software system.

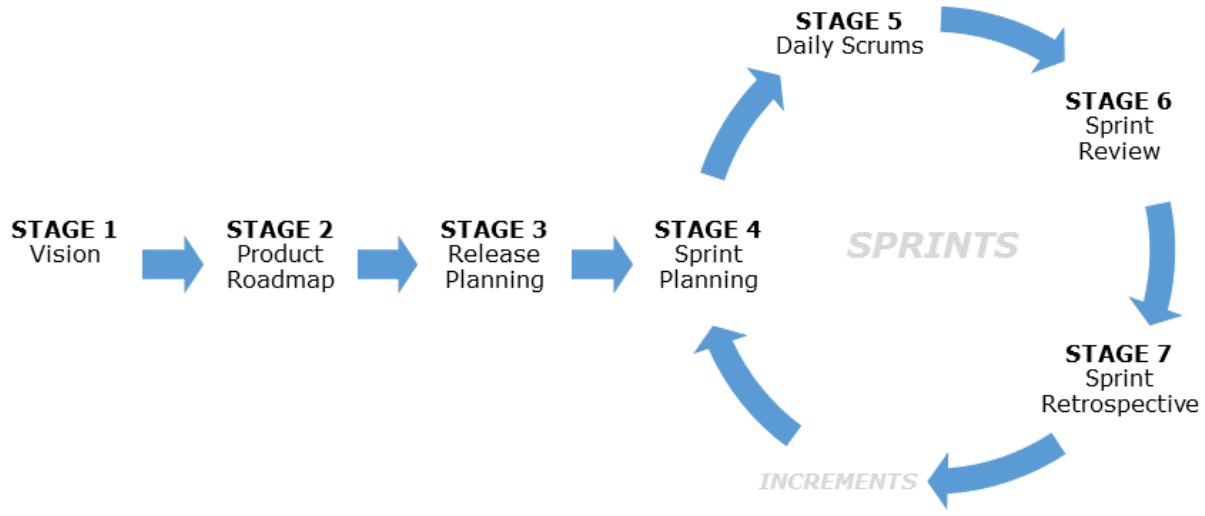
Objectives of software testing

- Preventing defects from entering the system.
- Finding defects existing in the system.
- Measuring the quality of the system.





Level	Objective	Test Basis	Done By
Component	Used to test the functional and non-functional characteristic of the code/module.	• Component Requirement • Code	White Box Testers
Component-Integration	Used to test the interaction between software components and is done after component testing.	• Software and System design. • Architecture • Use Cases	White Box Testers
System	Used to test the functional and non-functional characteristics of the complete system.	• System and Software requirement Specification. • Use Cases	Black Box Testers
System-Integration	Used to test the interaction between different systems or between hardware and software.	• System and Software requirement Specification. • Use Cases • Software and System design.	Black Box Testers
UAT – Alpha	Also called as factory acceptance testing. It is done at developing organizations site but not by the development team.	• System Requirements • User Requirements • Use Cases	Users / Stakeholders
UAT - Beta	Also called as field/site acceptance testing is performed by customers at their own locations.	• System Requirements • User Requirements • Use Cases	Users / Stakeholders

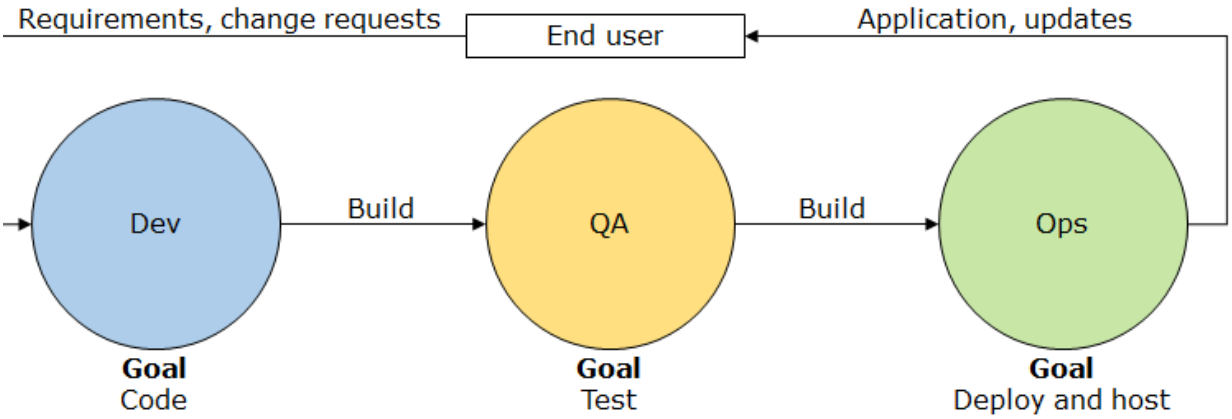


1.DevOps

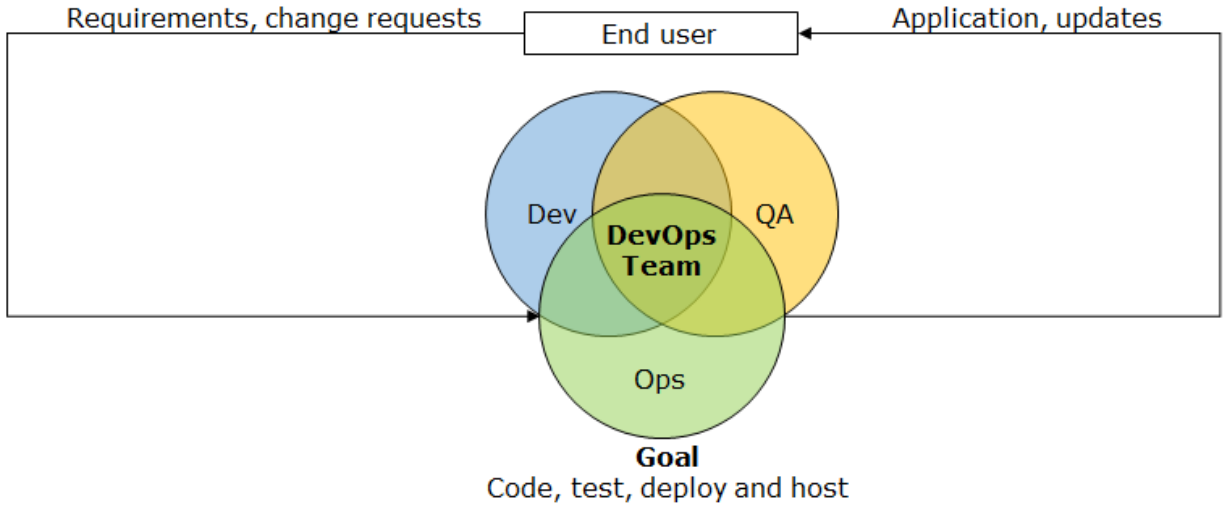
Yes. High deployment frequencies are possible and leading IT organizations do it today.

- **FlickR** - The popular image and video hosting portal deploys release updates to their applications everyday.
- **Facebook** - The most used social networking site, deploys an average of 2 releases everyday.
- **Amazon** - The world's largest internet company by revenue, does an average of 50,000,000 code deployments per year.

They do it with the help of DevOps.

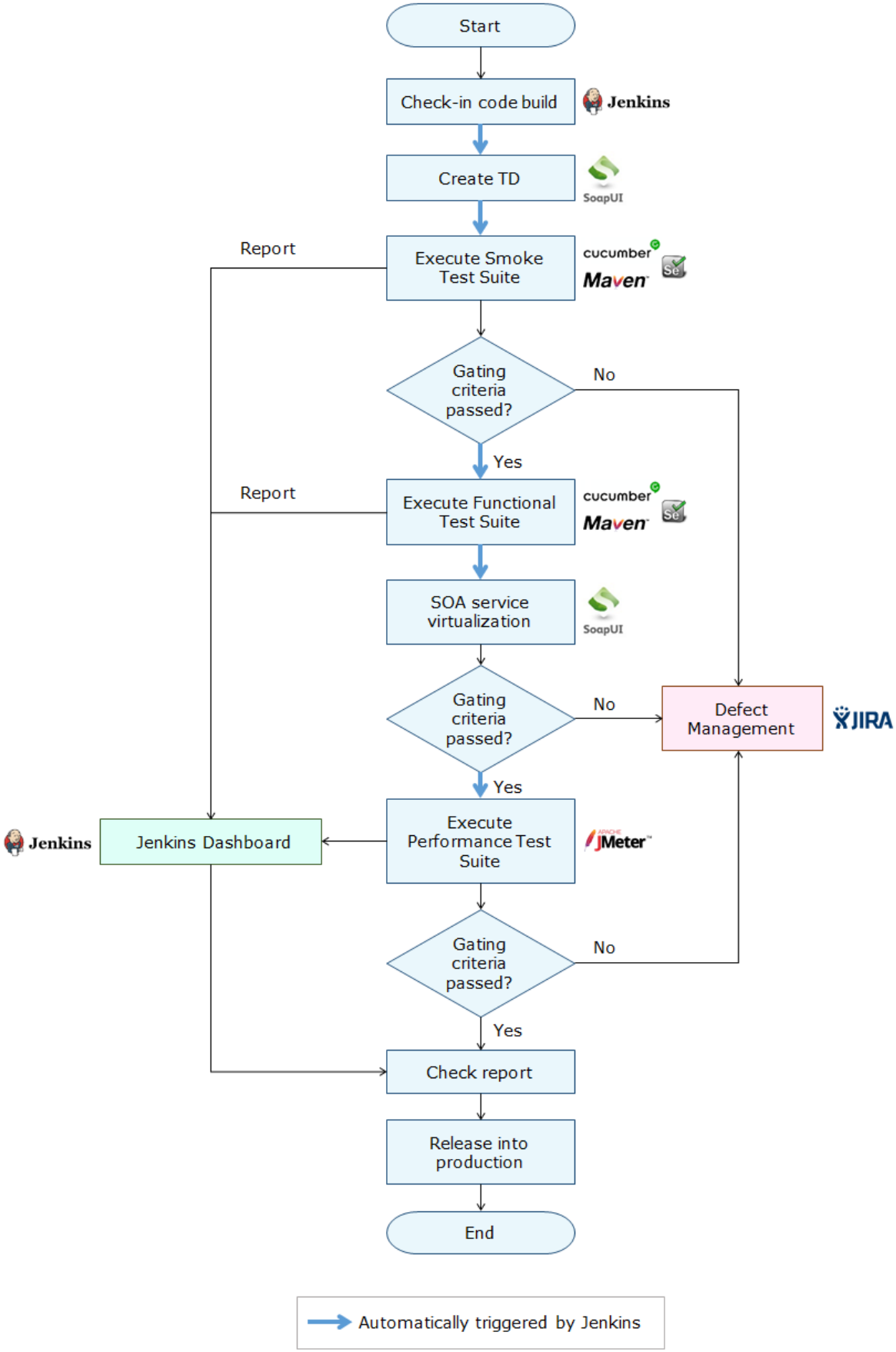


How DevOps teams work



Continuous Testing (CT): Continuous Testing is the process of executing automated tests as part of the software delivery pipeline in order to obtain rapid feedback on the business risks associated with a software release candidate

	Continuous Integration (CI)		Continuous Delivery (CD)	
Levels	**Build Management**	**Test Management**	**Environment Management**	**Release and Deployment Management**
Level 0	• Manual Builds • No build repository • Manual Quality Test	• Manual \ Ad hoc testing • Insufficient Test Env • Test strategy \Test cases • Defect management tool	• Manual environment provisioning • Inconsistent Env Config	• Unplanned and Infrequent Releases • Manual Deployment
Level 1	• Scripted Builds • Code Quality Checks through tool • Source Control and Versioning	• Test environment setup • Unit test automation	• Automated environment provisioning • Consistent Env Config	• Defined release procedure • Scripted Deployment Automation
Level 2	• Automated build, code quality checks invoked by CI and stored in build repository	• Automated regression test suite • System test automation • Performance test automation	• Version control for env setup and config • Environment monitoring tools and dashboard	• Release management automation using tool • Release mgmt. tool in CD pipeline to forward deploy / rollback
Level 3	• Triggered CI Builds • Gated auto build promotion through environments	• Test automation invoked by CI / release automation tool • Acceptance test automation • Automated test data replication	• Automated On demand environment provisioning • Infrastructure as code	• Release management integrated with infra as code
Optimized	• Build metrics gathered and analyzed • Parallel build management	• High test coverage Parallel testing • Integration with change and defect management tools	• Env metrics \ SLAs defined and Monitored • Environment dashboard / Advanced env diagnostics	• Release metric gathered and analysis • Performance analysis


Start
Check-in code build
Jenkins
Create TD
SoapUI
Report
Execute Smoke Test Suite
cucumber
Maven
Gating criteria passed?
No
Yes
Report
Execute Functional Test Suite
cucumber
Maven
SOA service virtualization
SoapUI
Gating criteria passed?
No
Defect Management
JIRA
Yes
Jenkins
Jenkins Dashboard
Execute Performance Test Suite
JMeter
Gating criteria passed?
No
Yes
Check report
Release into production
End
Automatically triggered by Jenkins

PipeLines :

international telecommunications and television company

In this case study we will look at how DevOps was implemented for an international telecommunications and television company. This client is one of the largest broadband internet service providers outside of the United States

XYZ Company helped them to successfully build overall testing governance structure using DevOps practices.

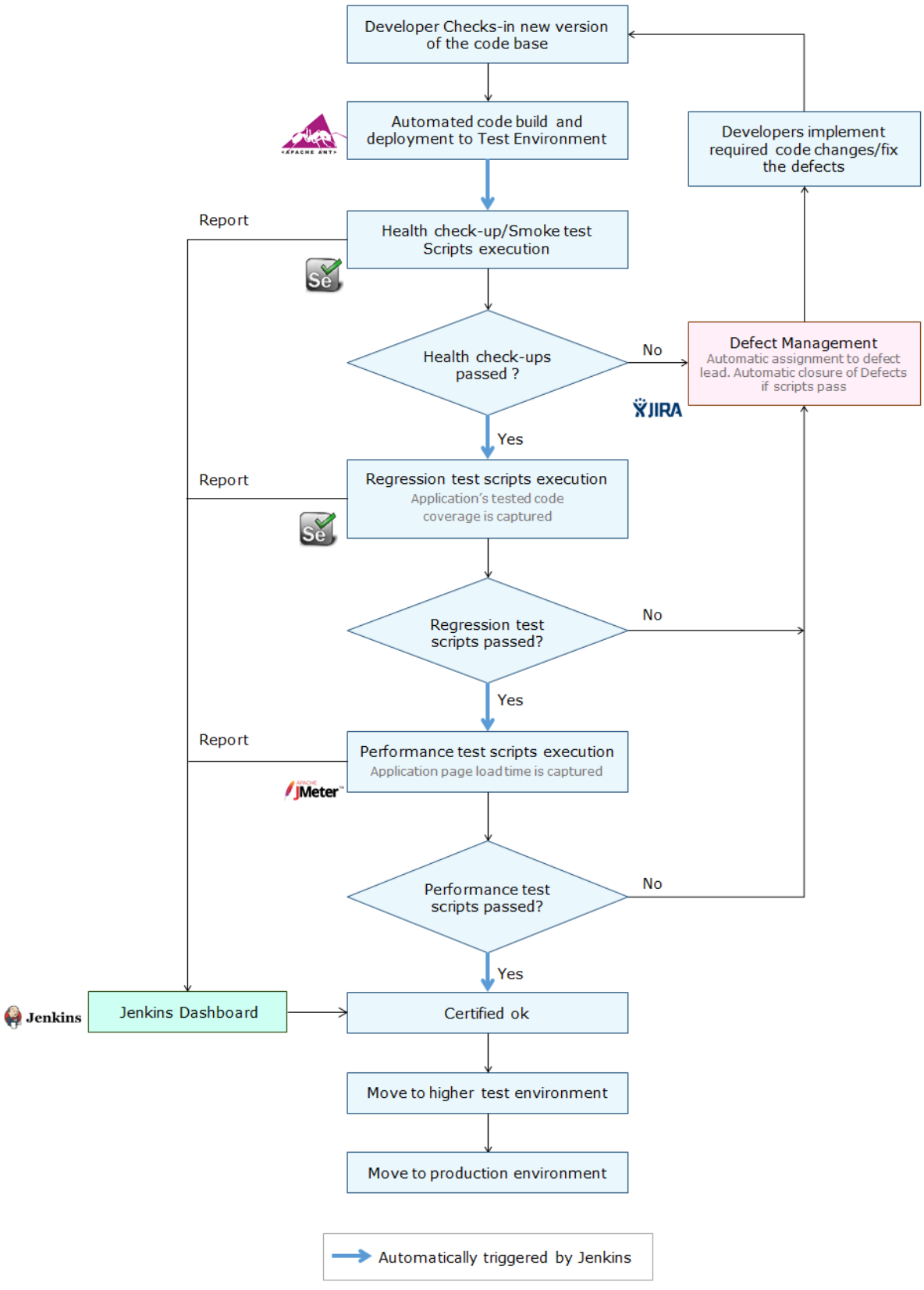
Before DevOps implementation:

With deployment frequencies as high as atleast 2 per week, it was challenging due to:

- Total integration time, from code deployment to test readiness taking 5 weeks
- Manual regression tests to test every deployment.
- Manual smoke tests required to ensure all the services are up and running on different environments.

Tools used:

- **Jenkins**: For automated triggering of CI-CD pipeline processes
- **Apache Subversion:** For automated version control of code base and automated test scripts
- **Apache Ant**: For automated software build processing
- **JIRA:** For defect management
- **Selenium:** For functional test automation
- **Apache Jmeter:** For performance testing


Developer Checks-in new version of the code base
Automated code build and deployment to Test Environment
Developers implement required code changes/fix the defects
Report
Health check-up/Smoke test Scripts execution
Health check-ups passed ?
No
Defect Management
Automatic assignment to defect lead. Automatic closure of Defects if scripts pass
JIRA
Yes
Report
Regression test scripts execution
Application's tested code coverage is captured
Regression test scripts passed?
No
Yes
Report
Performance test scripts execution
Application page load time is captured
Performance test scripts passed?
No
Yes
Jenkins
Jenkins Dashboard
Certified ok
Move to higher test environment
Move to production environment
Automatically triggered by Jenkins

Continuous integration: The most dominant player in the 'Three Cs' is Continuous integration (CI) and it's a necessary approach for any Agile team. CI requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.

By integrating regularly, teams can detect errors quickly, and locate them more easily. Simply, it ensures bugs are caught earlier in the development cycle, which makes them less expensive to fix - and maintains a consistent quality.

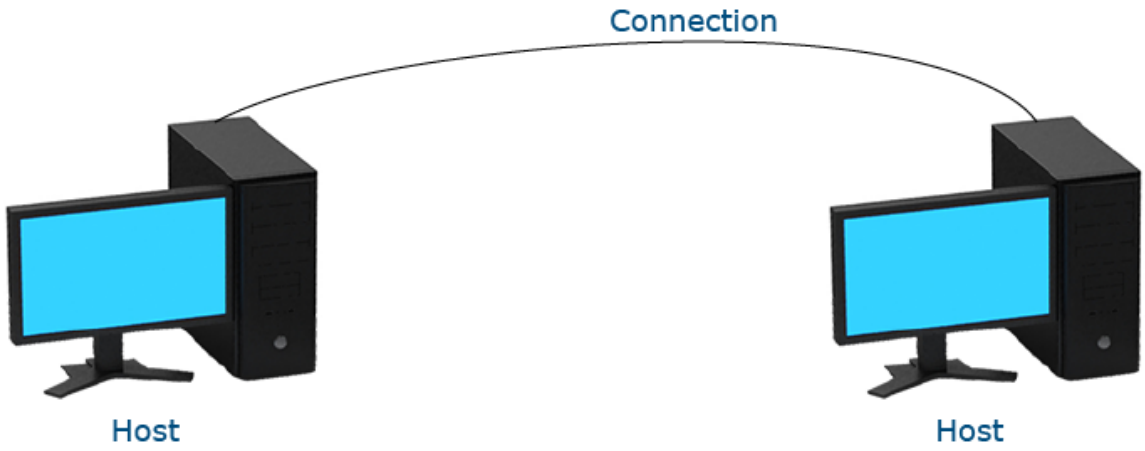
Continuous delivery: Continuous delivery is the practice of streamlining/automating all the processes leading up to deployment. This includes many steps, such as validating the quality of the build in the previous environment (ex.: dev environment), promoting to staging, etc. These steps, done manually, can take significant effort and time. Using cloud technologies and proper orchestration, they can be automated.

Teams should ensure they have a monitoring dashboard for your production environment in place in order to eliminate performance bottlenecks and respond fast to issues. This will complete an efficient CD process.

Continuous testing: Continuous testing (CT), which can also be referred to as Continuous Quality, is the practice of embedding and automating test activities into every "commit". CT helps developers use their time more efficiently when trying to fix a bug for code that was written years ago. To fix the bug, developers should first remind themselves of which code it was, undo any code that was written on top of the original code, and then re-test the new code; not a short process. Testing that takes place every commit, every few hours, nightly and weekly, not only increases confidence in the application quality, it also drives team efficiency.

2. Networking

When two or more machines are connected with each other, it is called a network and the devices in a network are called hosts.



Your classroom computer is actually part of XYZ Company network. All the computers in XYZ Company are connected with each other, That's why you are able to send a mail to any XYZ Company mail Id.

What do you think is the use of LAN cable?

LAN cable is used to transmit data to and from the computer.

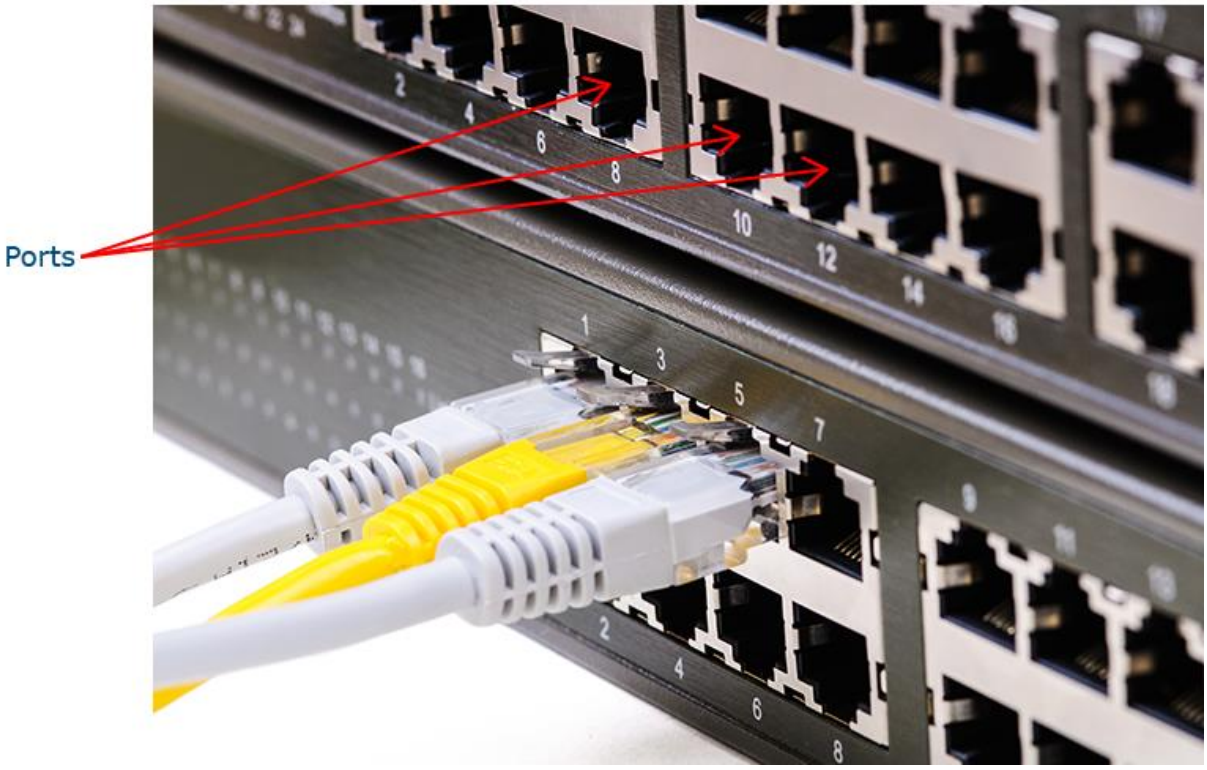
The amount of data that can be transmitted in a given period of time is called Bandwidth. It is measured in Mbps, Gbps, etc.

Type of cable	Bandwidth
Twisted-Pair	100 Mbps
Fiber-Optic	>10Gbps

The LAN cable starts from your desktop. Do you know where it ends?

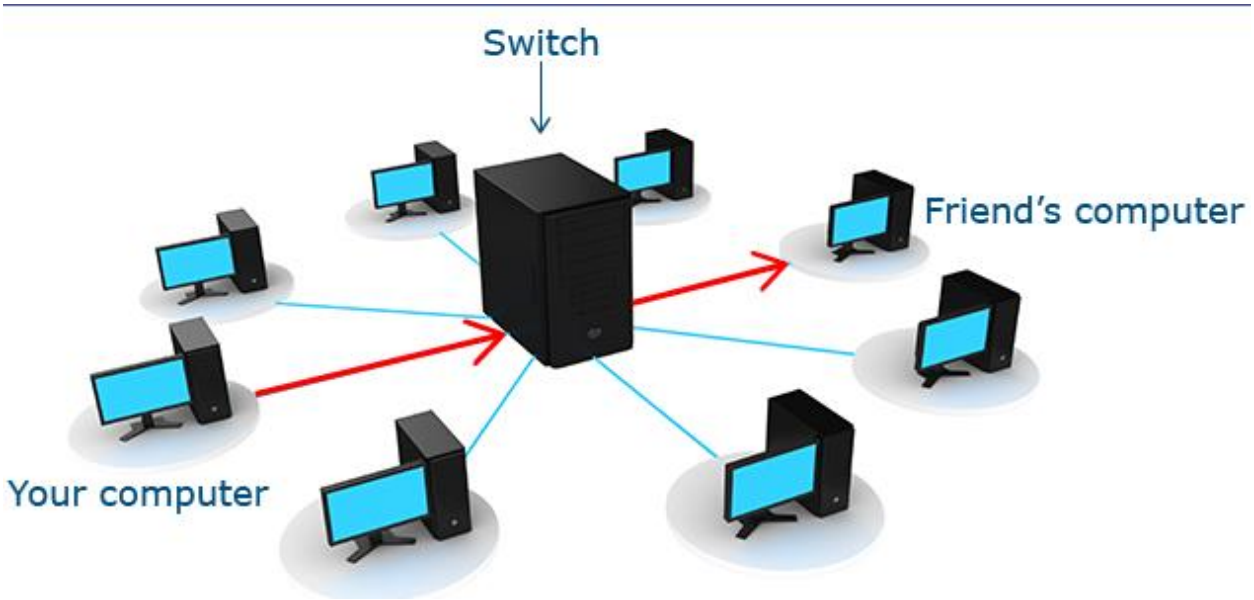
It ends in a Switch.

Switch has many ports. Each port can be connected with an individual network device



Check if your computer is directly connected to your friends computer. How does your data reach his computer?

- The message you send reaches the switch. The switch sends the message to your friends machine.
- A switch is a device which connects all the devices within a network. All your computers are connected to a switch!



Here we can see that all the computers are connected to a single switch, like a star. The layout of a network is called a topology and local networks use the star topology.

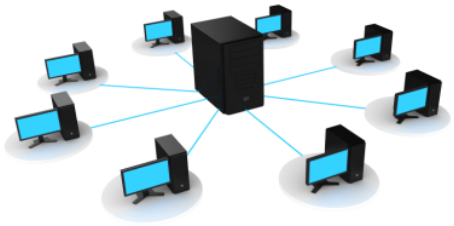
Infosys Mysore network

Super-market network within Infosys Mysore campus

Different devices are grouped to form different networks. Why do we need different networks? Why not just one big network of devices?

Control:

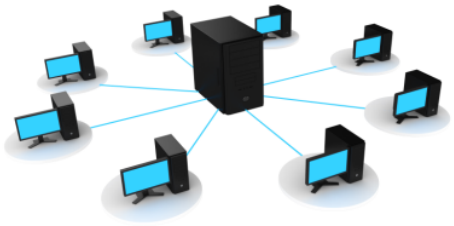
No Internet connection before 5 P.M.
Only Infosys employees can access network



Infosys Mysore network

Control:

24 hours Internet connection.
Only store employees can access network

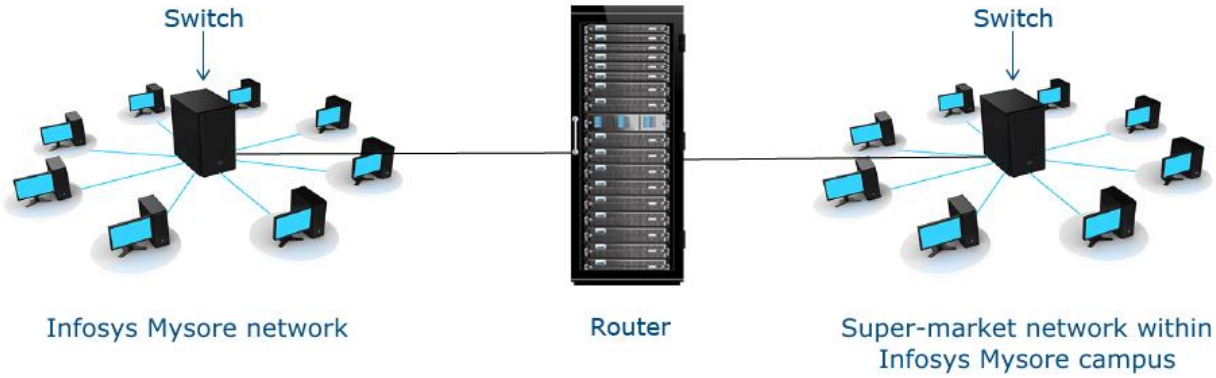


Super-market network within Infosys Mysore campus

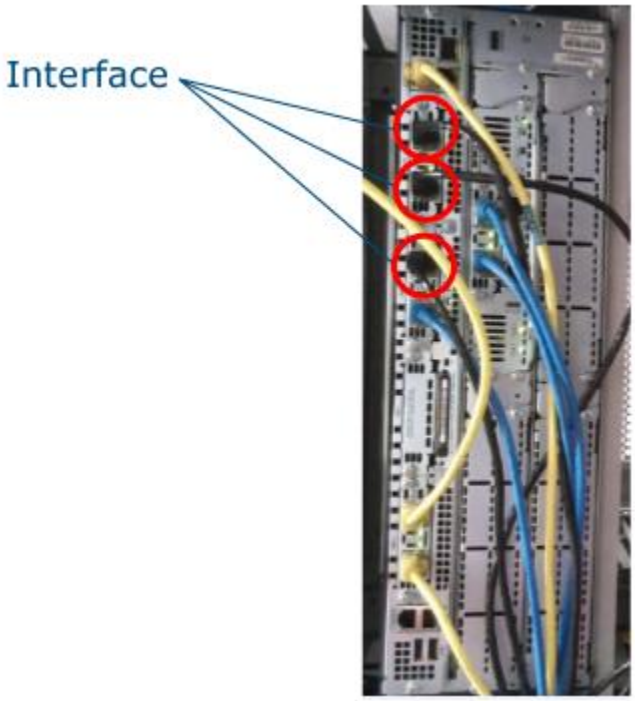
We have different networks so that we can control them differently.

If your switch cannot send data beyond XYZ Company Mysore network, how can you send data to a machine in the supermarkets network?

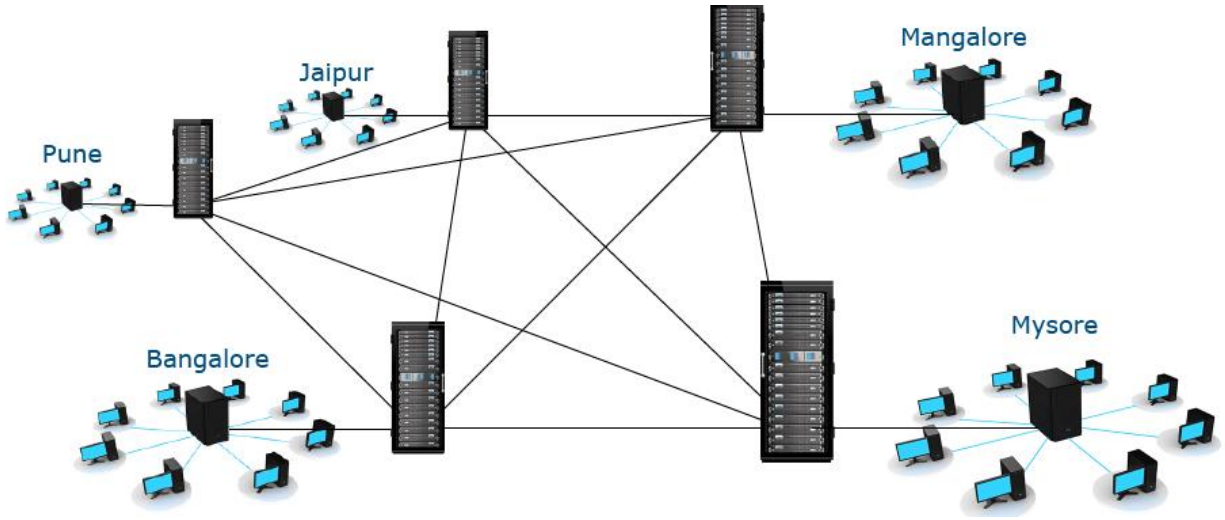
A router is a device which is used to connect different networks.



Just the way a switch has ports, a router has interfaces through which other switches and routers are connected.



We can form a network of networks also. Internet is the largest network of networks!

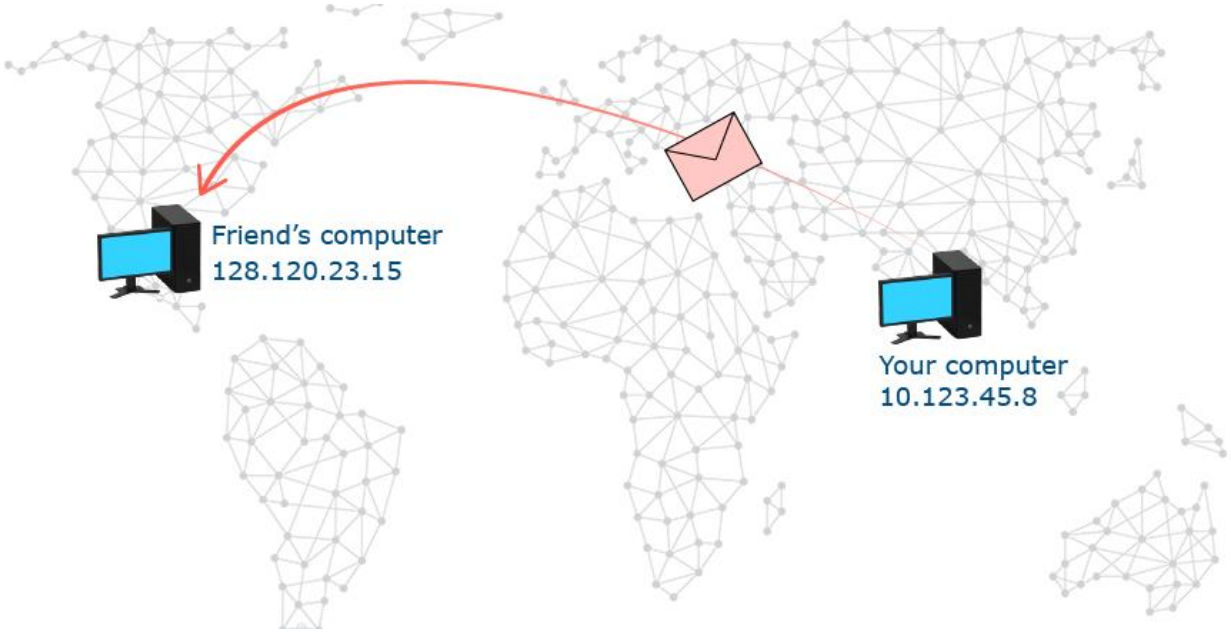


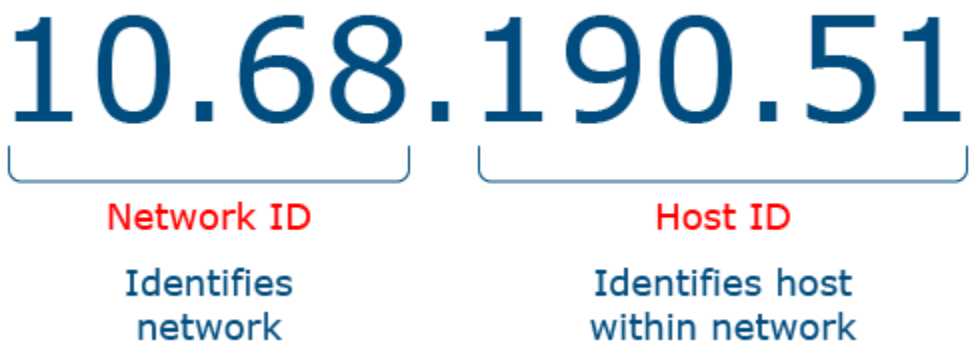
Here we can see that the routers are highly interconnected, like a mesh. Such a topology is called a Mesh topology. The mesh topology improves redundancy, as the data can reach the destination in a different route if some link fails.

IpAddress

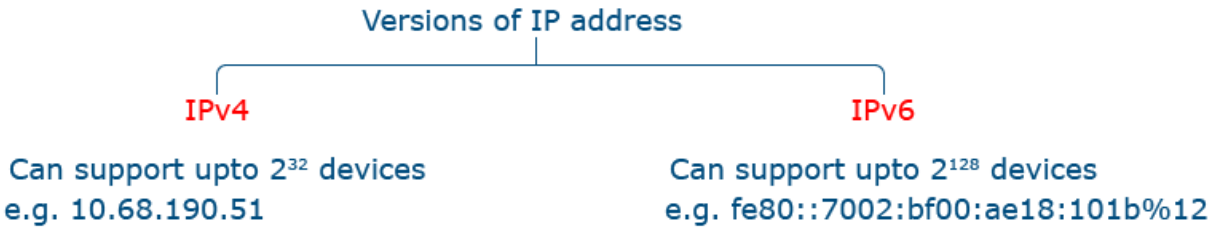
Let's say you want to send a parcel to your friend in USA. There are millions of houses all over the world. How can you uniquely identify your friends house?

Just the way you can uniquely identify your friends **house by the unique house address,** we can uniquely **identify every device in the network by its address called the IP address!**





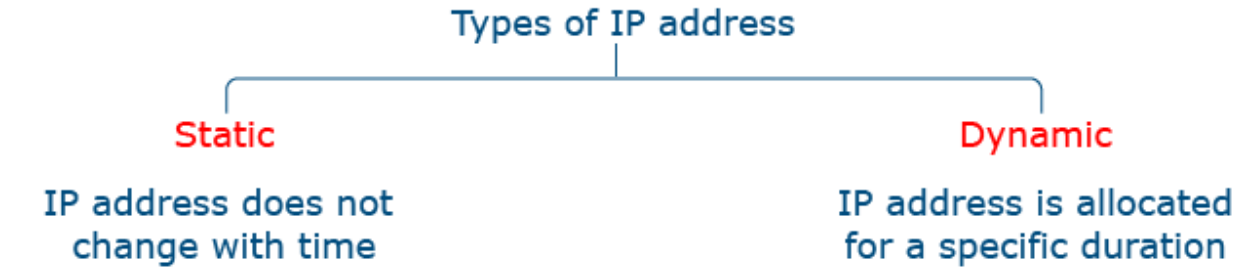
Since the number of devices on the internet far exceeds the number that can be supported by IPv4, world is gradually adopting the IPv6 system



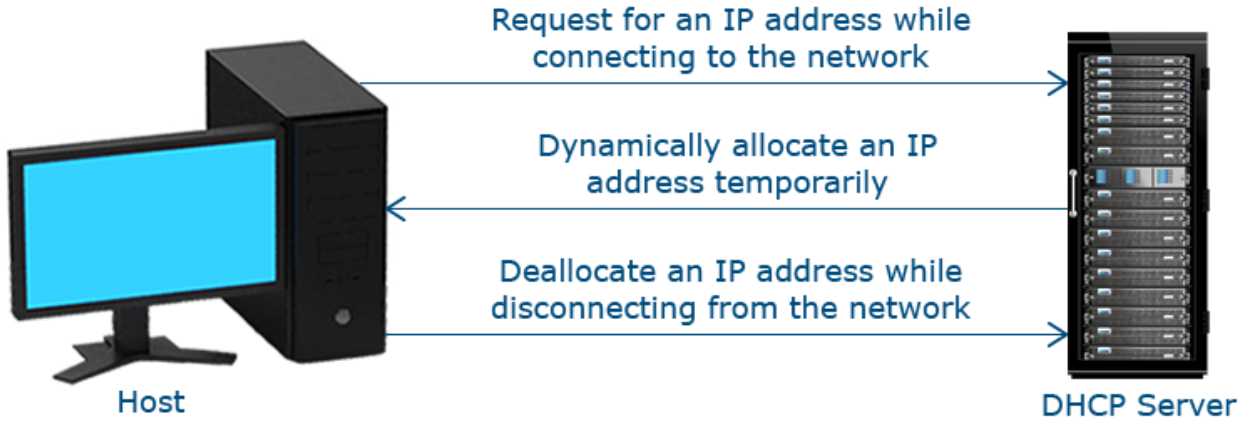
Find your IP address by typing ipconfig -all in command prompt.

Since available IP addresses are limited, organizations allocate IP addresses to machines temporarily and later deallocate them. A machine which does this is DHCP server. DHCP

stands for Dynamic Host Configuration Protocol.

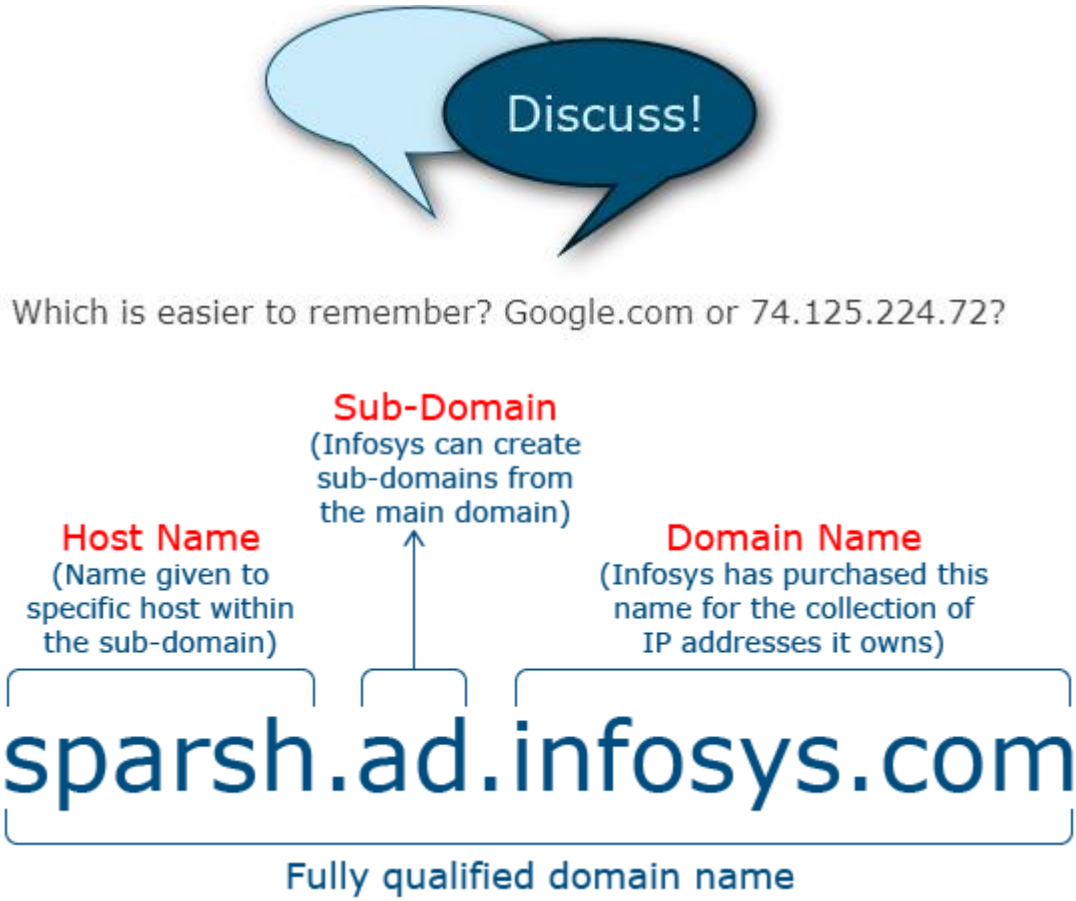


The below approach is used for trainee machines.



Find the duration of the temporary IP allocation through ipconfig -all

```
Connection-specific DNS Suffix  . : ad.infosys.com
Description . . . . . . . . . . . : Realtek PCIe GBE Family Controller
Physical Address. . . . . . . . . : 10-62-E5-02-BB-42
DHCP Enabled. . . . . . . . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::6c72:54d:6d62:4fb9%4(Preferred)
IPv4 Address. . . . . . . . . . . : 10.219.19.81(Preferred)
Subnet Mask . . . . . . . . . . . : 255.255.255.0
Lease Obtained. . . . . . . . . . : Wednesday, September 12, 2018 11:17:09 AM
Lease Expires . . . . . . . . . . : Wednesday, September 12, 2018 1:17:09 PM
Default Gateway . . . . . . . . . : 10.219.19.1
DHCP Server . . . . . . . . . . . : 10.219.3.103
DHCPv6 IAID . . . . . . . . . . . : 51405541
DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-22-AB-48-DB-10-62-E5-02-BB-42
DNS Servers . . . . . . . . . . . : 10.219.120.5
                                   10.136.65.55
                                   10.136.65.42
NetBIOS over Tcpip. . . . . . . . : Enabled
```



Domain Name is a name given to an IP address or a collection of IP addresses. No two organizations can have the same domain name.

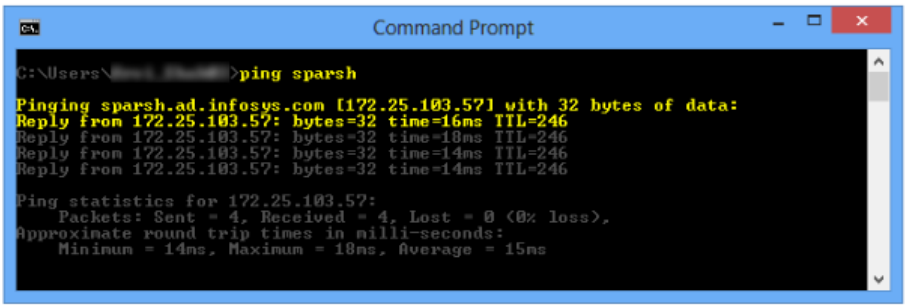
www.XYZ Company.com is the name given to the IP address of the computer within XYZ Company.com domain, which stores the XYZ Company website

DNS (Domain Name System) Server is a machine which has a database of domain names and the corresponding IP addresses.

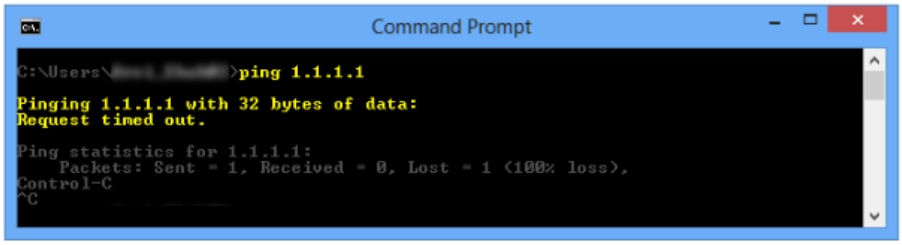
DNS is used for Resolving names into IP address

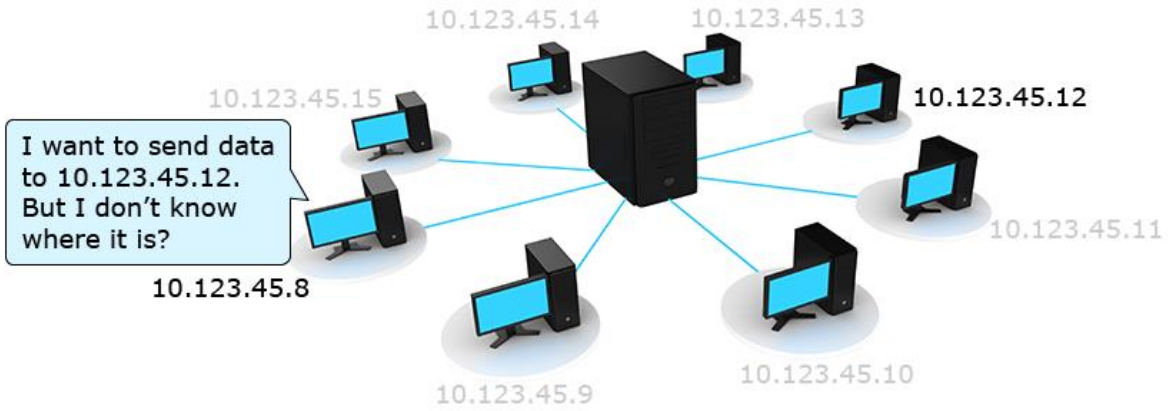
Ping is a command which checks the connectivity with a specific machine. It also gives the IP address associated with a domain name

Ping - Successful



Ping - Unsuccessful





Your machine will check if the destination machine is within the same network.

How can your device know if the destination IP address is within the same network?

If the network ID of the IP addresses are same, then it means both the addresses are in same network.

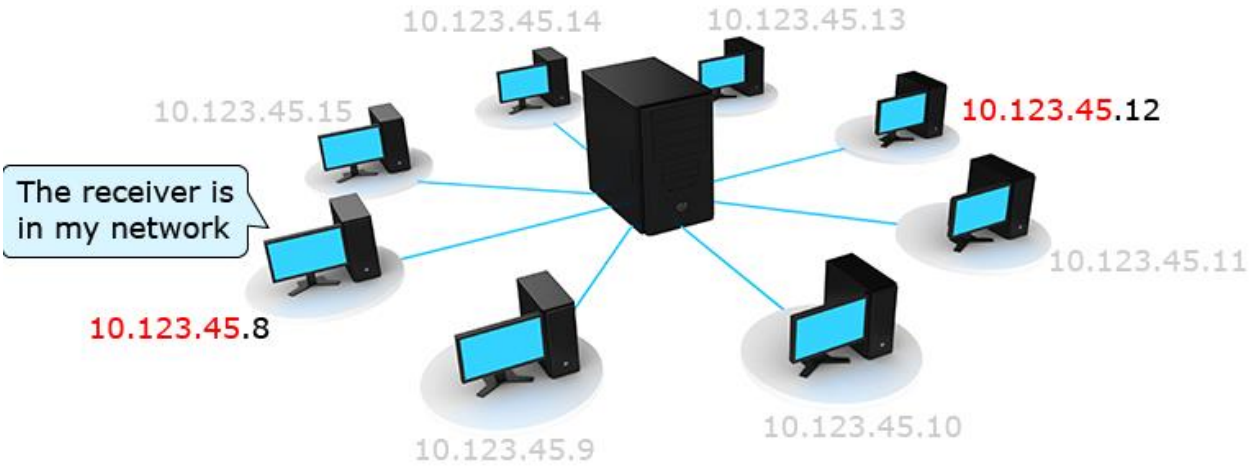
Subnet mask is a value which is used to separate out the network ID and the host ID.

For example, consider IP address 10.68.190.51

Subnet mask	Network ID
255.0.0.0	10
255.255.0.0	10.68
255.255.255.0	10.68.190

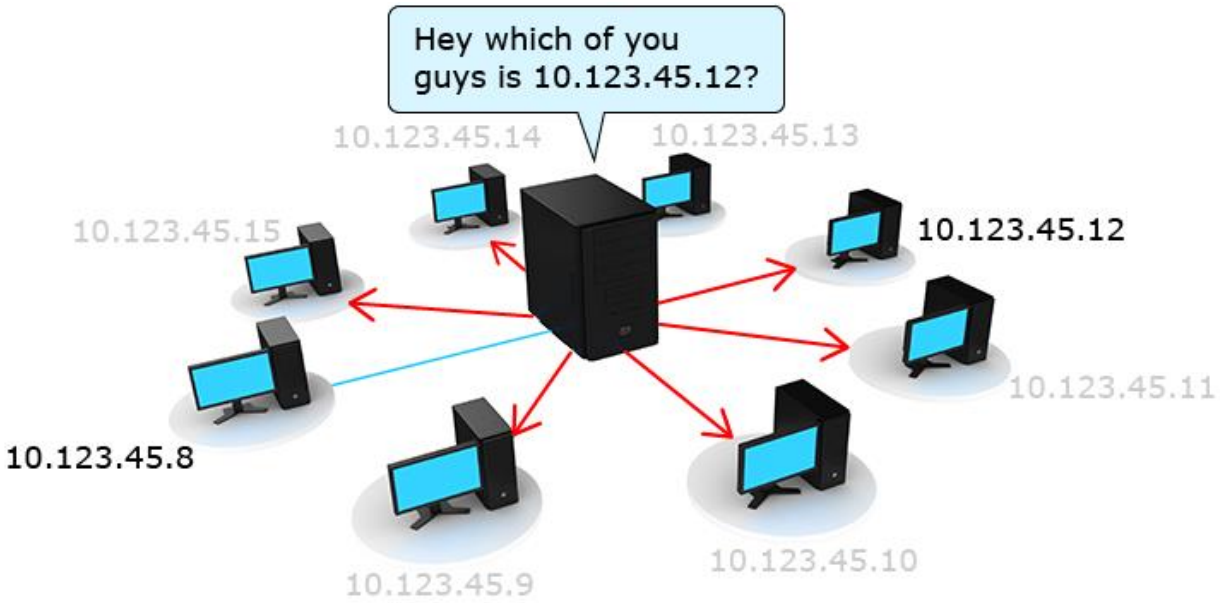
Note: Network ID is determined by a bitwise AND operation between IP Address and Subnet Mask.

Applying the subnet mask of 255.255.255.0, the Network ID of sender and receiver is 10.123.45. Hence, they both are in the same network.

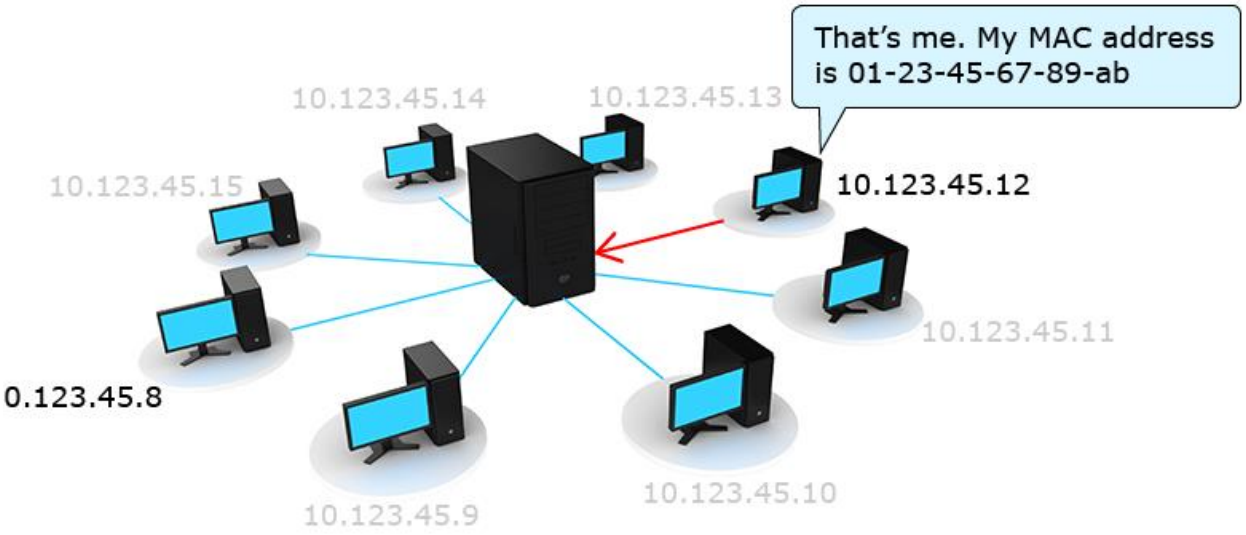


Among all the machines in the same network how can your machine find out which is the receiver machine?

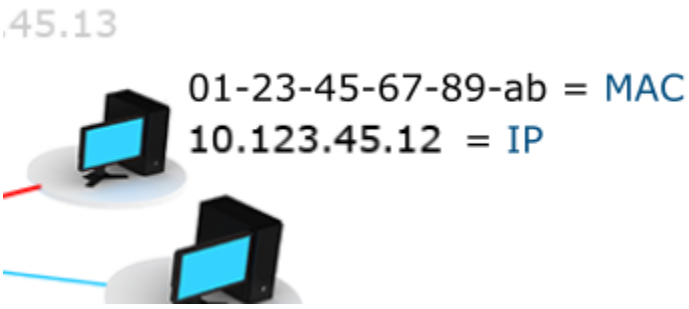
A similar thing happens in a network also. Switch sends a broadcast message to the rest of the devices



The machine with the IP address responds back, giving details of its MAC address. MAC is a unique value given to a
computer.



This process is called ARP. ARP – Address Resolution Protocol
MAC – Media Access Control
A MAC address is also a unique value given to every device.



If IP address itself can uniquely identify a computer, then why do we need a MAC address?

Different unique values are meant for different purposes. For example, your employee number though unique is relevant only within XYZ Company, whereas a unique passport number is relevant throughout the world.

Similarly, IP and MAC are meant for different purposes.

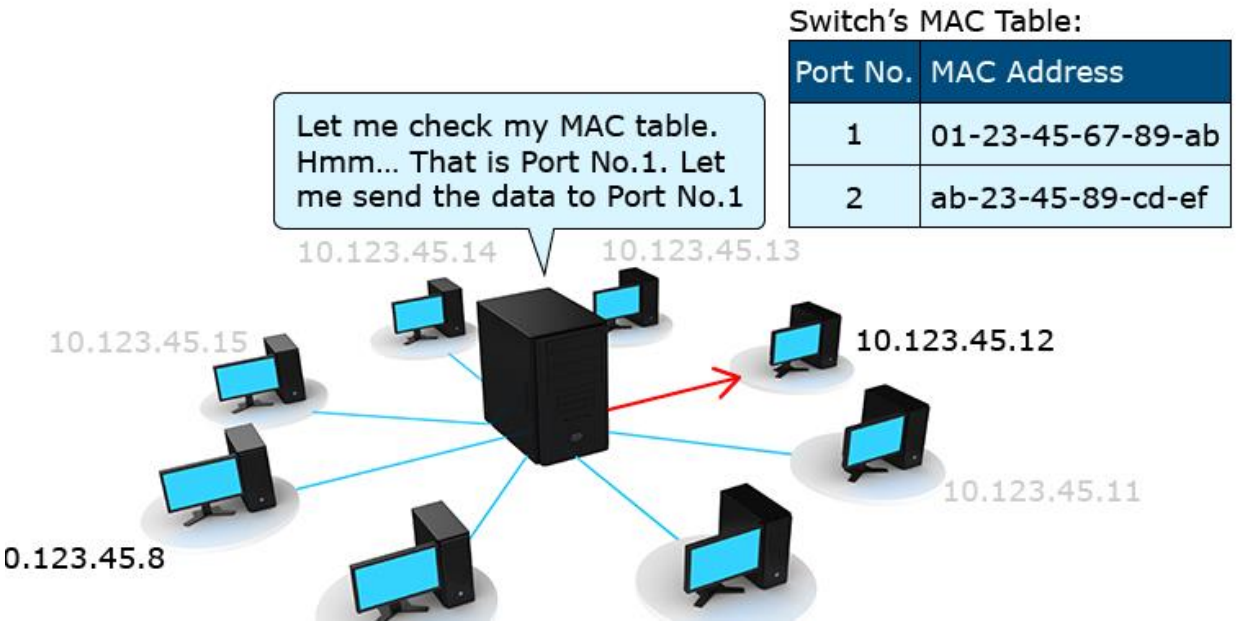
Some uses of Mac are:

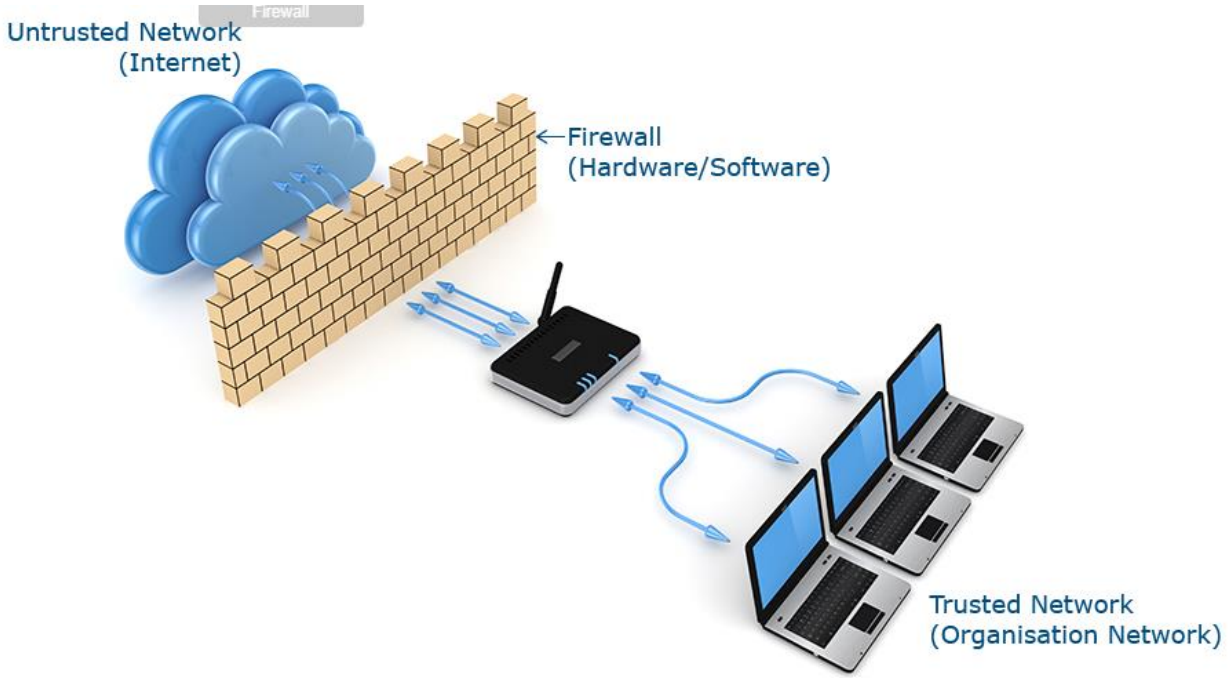
- Uniquely identify a machine when IP address is not available. For example, a DHCP server uses MAC address to assign the IP address.
- A IP address can change, for example, when you take your laptop from one DC to another. **But, MAC is permanent**. **That is why it is also called the physical address of the system.**

IP Address	MAC Address
Unique	Unique
Example: 10.123.45.12	Example: 01-23-45-67-89-ab
It can change.	It is permanent. It never changes.
Used to identify the network and the device	Used to identify the device
Switch does not understand IP address	Switch understands only MAC address and PORT.

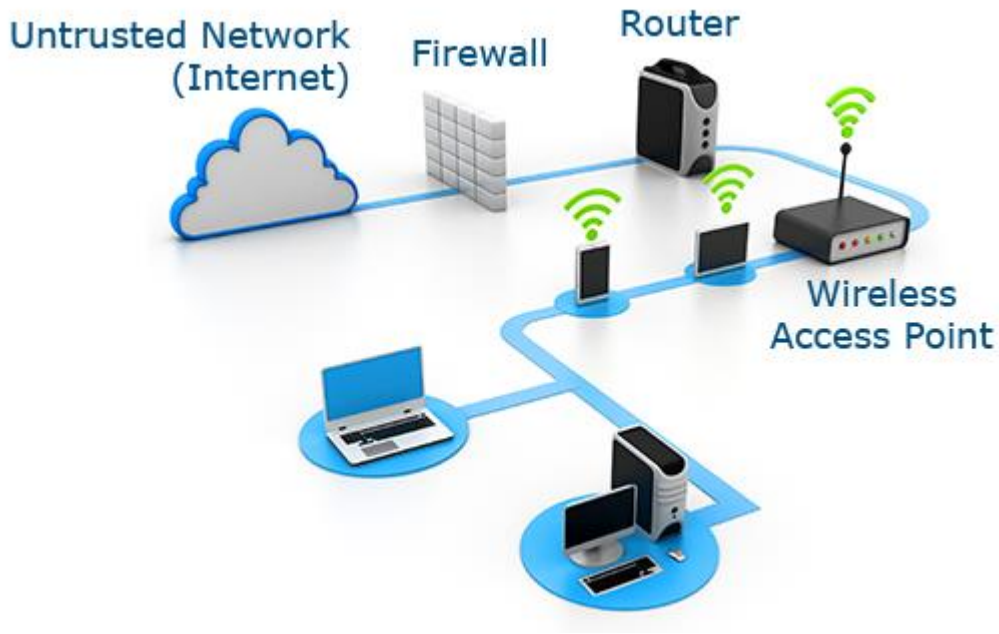
Now that your machine knows the MAC address of the receiver machine, it can now send the data.

The switch has a MAC table. The MAC table has a list of port numbers and MAC addresses. Depending on the MAC address passed, it will send information to that specific machine alone.





Your mobile is not connected to network through wires. Then how it is able to send and receive data? This is done through Wi-Fi.



Let's say that you are browsing Sparsh.

By the time the Sparsh page loads in your screen, the data has gone through so many software and hardware!

Layers	Typical Protocol stack	
	For sending a email	For viewing a video
Application Layer	SMTP	HTTP
Transport Layer	TCP	UDP
Internet Layer	IP	IP
Link Layer	Ethernet	Ethernet

Start the web service from your command prompt
python CurrencyConverterService.py

Call the web service from your web browser
http://127.0.0.1:9999/currency_convert?amount_in_dollars=6

Open another command prompt. Call the web service using python client
python CurrencyConverter_Python_client.py 6

Open another command prompt. Call the web service using java client
java CurrencyConverter_Java_Client 6

Download Demo

Note 1: Ensure you are in the correct directory in the command prompt.

Note 2: To stop the server, press ctrl+c in the command prompt

3.Cloud Computing

Cloud computing means that all the computing hardware and software resources that you need to process your tasks are provided for you, "as a service" over the internet, by a vendor instead of you owning and maintaining them.

It is the responsibility of the vendor, the **Cloud Service Provider (CSP)**, to develop, own and maintain these resources and make them available to the consumers over the internet.

You, the consumer, need not know exactly where the resources are located and how it all works.

Example
If you want to use an email service, you would need the hardware and software resources for

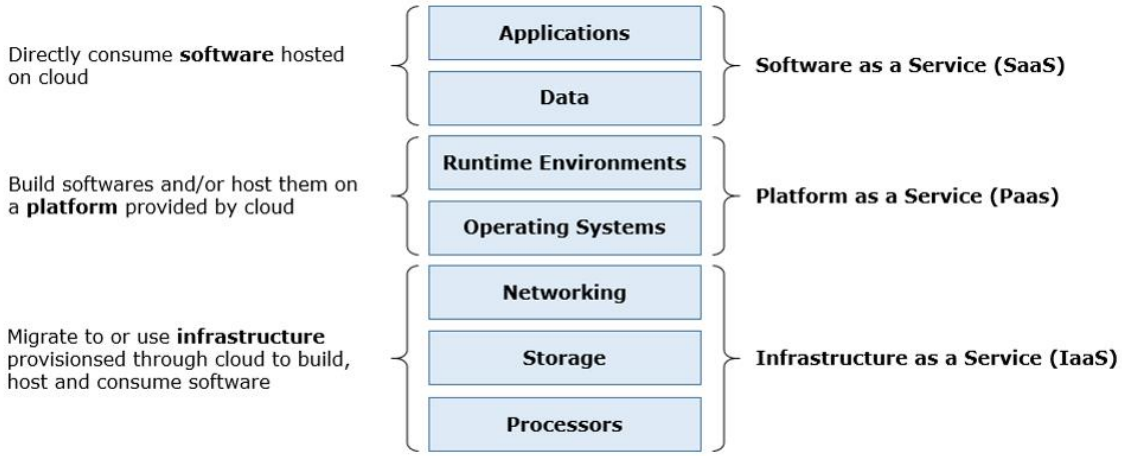
- an email server to send, receive and store your mails
- an email client to access the data and operations in your email server.

Instead, if you use a cloud based mail service like Gmail, Outlook, etc., all you need is a device, with an app or a browser, connected to the internet

A traditional enterprise IT set up (on-premise) would consist of the following layers.

Applications	To perform daily business operations. Eg: HR systems, Outlook, CRMs, etc.
Data	Generated and used by business applications. Eg: documents, emails, files, etc.
Runtime Environments	To enable building and executing software programs in specific technologies. Eg: JRE, interpreters, compilers, etc.
Operating Systems	To enable users and applications to communicate with the hardware. Eg: Windows, Linux, Solaris, etc.
Networking	To enable communication between all the hardware and software components. Eg: Routers, switches, gateways, protocols, etc.
Storage	To hold applications and the data that they process. Eg: Hard disks, tape storages, flash memory, etc.
Processors	To perform the actual computing. Eg: Intel Xeon, AMD Opteron, Qualcomm Centriq, etc.

Organisations are looking at cloud computing solutions like below



1.Infrastructure as a Service is the provisioning of IT infrastructure resources, like processors, storage, networks, firewalls, load balancers etc., over the internet.

Amazon Simple Storage Service (S3)

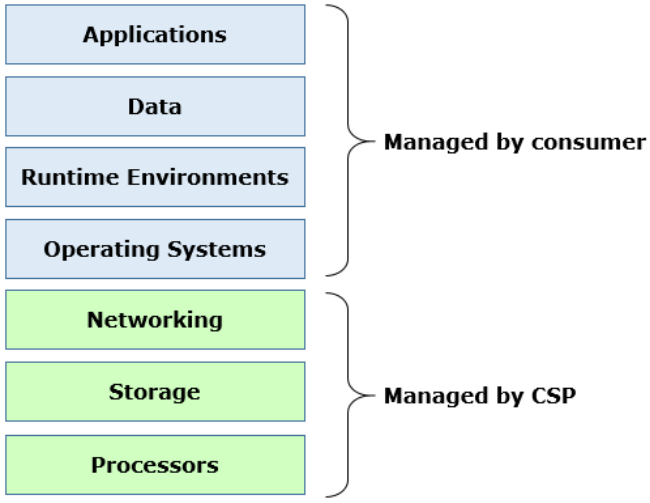
Developers looking for highly scalable, reliable, fast, inexpensive data storage infrastructure can opt for Amazon's cloud based storage solution, the S3.
It has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web.

Google Compute Engine

Consumers of the Google Compute Engine can use virtual machines (VMs) hosted on the same supporting infrastructure that Google uses internally for end-user products like Google Search, YouTube, GMail etc.
Users can create the VMs from the standard images provided by Google or they can create custom images based on their infrastructure requirements.

Citrix XenServer

IT administrators looking to host, deploy and manage virtual machines would benefit from using Citrix XenServer.
It is a cloud-based server virtualization and management solution, designed to work specifically with Citrix Systems' XenApp and XenDesktop tools for application and desktop virtualization respectively.



2.Platform as a Service provides all of the capabilities that you need to support a complete application lifecycle - building, testing, deploying, managing, and updating – using the same integrated environment.

Microsoft Azure

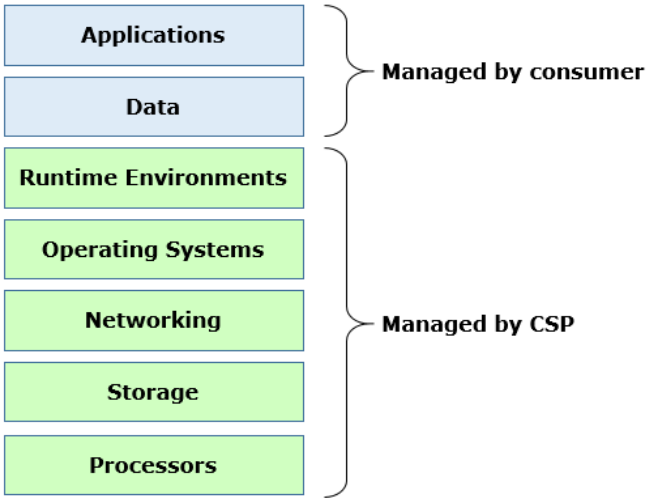
Microsoft Azure is a cloud computing platform which allows developers to build and deploy software, using tools, applications, and frameworks provided on it. The developer need not buy and maintain the underlying infrastructure, or manage the tool licenses and software upgrades.

Google App Engine

Google App Engine is a cloud-based platform which provides developers with popular built-in services and APIs for building web applications and mobile backends. The App Engine will also scale hosted applications automatically, in response to the amount of traffic it receives.

IBM Bluemix

IBM Bluemix provides a cloud platform that supports several programming languages and services, as well as, integrated DevOps to build, run, deploy and manage applications on the cloud. Developers using the platform also need not be concerned about installing softwares or having to deal with virtual machine images or hardware as these can be provisioned on the platform with just a few clicks or keystrokes.



3.Software as a Service describes any cloud service where a fully functional and complete software product is delivered to users over the internet. Instead of installing and maintaining software, you simply access it via the Internet, freeing yourself from complex software and hardware management.

LinkedIn

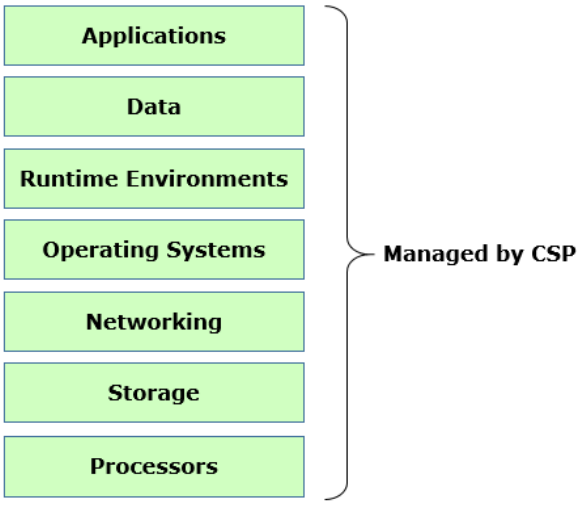
LinkedIn, a business and employment-oriented social networking service, offers three products: Talent Solutions, Premium Subscriptions and Marketing Tools.
Recruiters buy Talent Solutions to find, connect with and acquire talented people. Salespeople buy Premium Subscriptions to network and search on the platform. And businesses buy ads to market their products on LinkedIn.

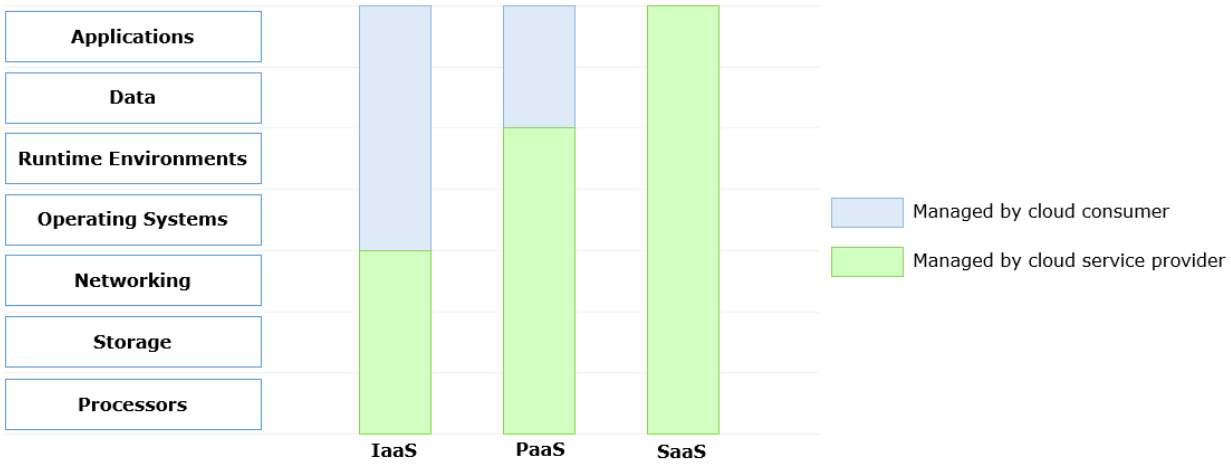
Salesforce CRM

Enterprises can use Salesforce's cloud based CRM software to
- streamline and automate tracking and managing of customer information
- analyze customer data to provide business insights and recommendations

Pixlr

Pixlr is a cloud-based photo editor that offers a wide range of image tools and utilities. It is like the traditional full version of Photoshop running on a web browser. It also offers a paid membership edition called Pro, which offers members exclusive access to advanced editing features.





Types of Cloud Platform plans

Each deployment model aims at addressing one or more concerns of the cloud consumer. Therefore, it is very important that consumers prioritize their concerns before opting for a particular model.

1. Public cloud
2. Private cloud
3. Community cloud
4. Hybrid cloud

1.Public Cloud is one that is available for use by the general public. Hence, it is the most common and popular deployment model available today. They are entirely owned, deployed, monitored, and managed by the cloud service provider, who deliver their computing resources over the internet.

Example

- DropBox provides storage space to the general public.
- Google provides Gmail and other cloud servers to the general public

2. Private Cloud is available only to users within a single organization.

Concerns addressed

- Security
- Compliance
- Governance/Control
- Performance

3.A community cloud is a private cloud that is shared by two or more organizations having shared concerns like security requirements, policy, and compliance considerations etc.

4.A hybrid cloud is a combination of two or more different cloud deployment models.

In Real-World

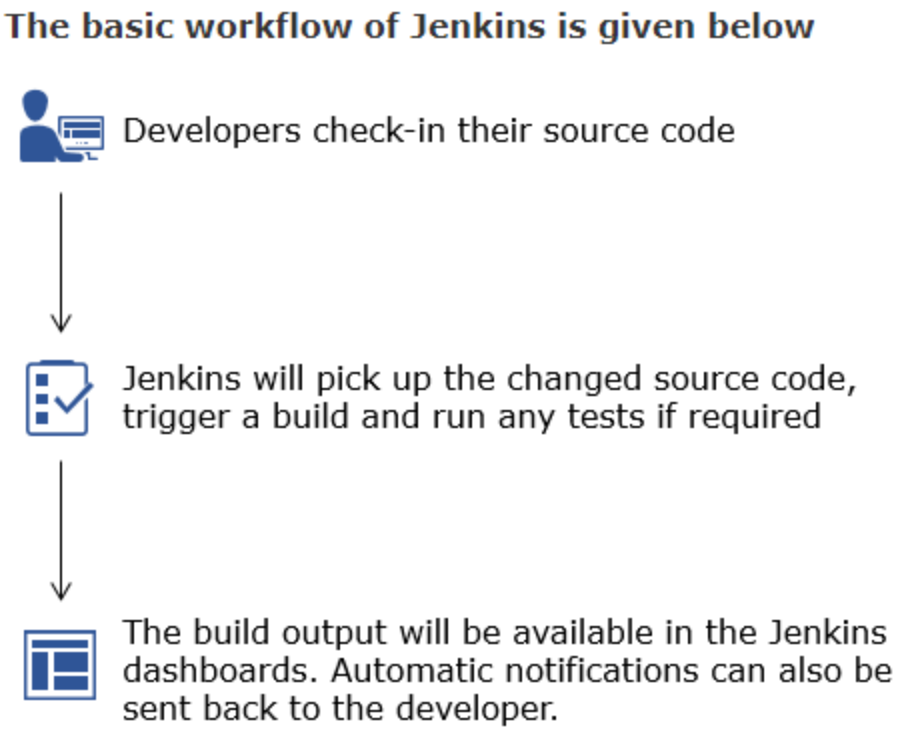
As a software services professional, you might get to work on cloud computing in one of the following ways

1. Cloud implementations
2. Cloud based developments
3. Migration projects

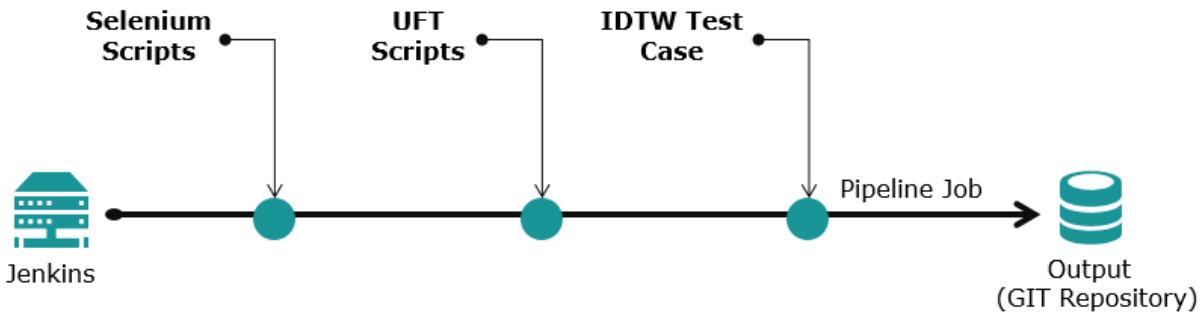
Misconception 3: Cloud computing is the same as virtualization

Though virtualization enables logical (not physical) separation of shared hardware resources, it is only an enabler for implementing cloud and not a mandatory requirement. Physically separate resources, like different makes and models of mobile devices, can also be hosted on cloud, without virtualization, to be used by developers and testers.

4.Jenkins

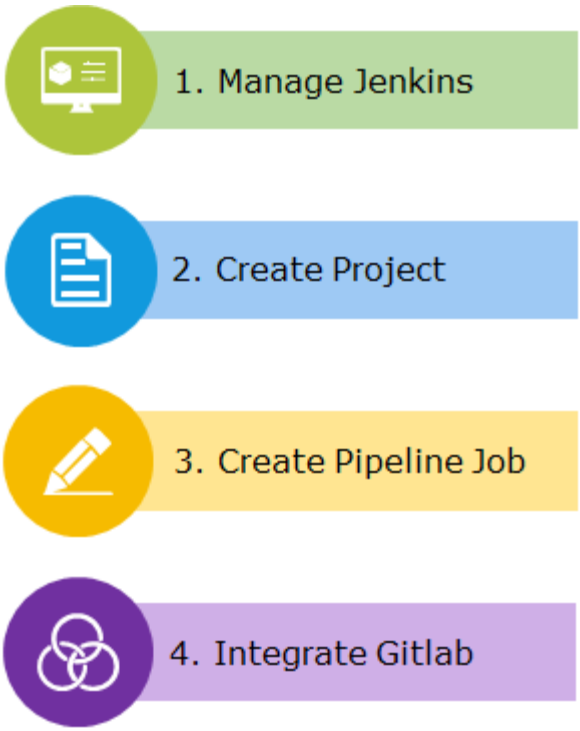


whenever developers create a build, testers will execute these test cases and scripts using its own frame works and the results will be saved separately for UFT, Selenium and IDTW.



With the help of Jenkins, we can automatically call and execute all these types of test cases and test scripts in a sequential or pipeline fashion whenever the developer pushes the code to GitHub. Also, the results of all these scripts can be saved back to GitHub. This helps in the automatic execution of test cases without the intervention of testers, and developers can immediately know the quality of their code.

```
cd D:\DevOps
d:
java -jar jenkins.war
admin /admin
```


1. Manage Jenkins
2. Create Project
3. Create Pipeline Job
4. Integrate Gitlab

5.Vagrant

"Create and configure lightweight, reproducible and portable environments."

Vagrant - the command line utility for managing the lifecycle of virtual machines

Vagrant, an open-source software product for **building and maintaining portable virtual development environments. Written in: Ruby**

The Hashicorp Repository Contains More Than 10,000 Boxes !

Alternatives: **Docker, CLI Tools, Terraform**

Vagrant is a tool for building and managing virtual machine environments in a single workflow. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time, increases production parity, and makes the "works on my machine" excuse a relic of the past.

Introduction

Every developer has faced problems when it comes to setting up a development environment. Usually the environment behaves as it should on one machine, while on another machine it behaves differently or does not function at all.

Vagrant changes the way how developers setup their work environments and maintain them. Vagrant makes it possible to create a configurable portable development environment easily. These environments need to be configured in a so-called **Vagrantfile** to get completely configured.

In this **Vagrantfile** file, the developer specifies how the environment should be set up, configured, which software should be installed and which operating system should be used. This Vagrantfile can then be distributed among other developers who just need this file in order to set up the same development environment on their own machine. Vagrant will then follow every step as defined in the provided Vagrantfile and initialise the machine.

- Vagrant encourages automation to set up your development environments using shell scripts or configuration management software.
- Vagrant allows you to work with the same operating system that is running in production, whether your physical development machine is running Linux, Mac OS X, or Windows.

If you were to run the virtual development environment manually — without Vagrant's help, that is — you would have to follow these steps:

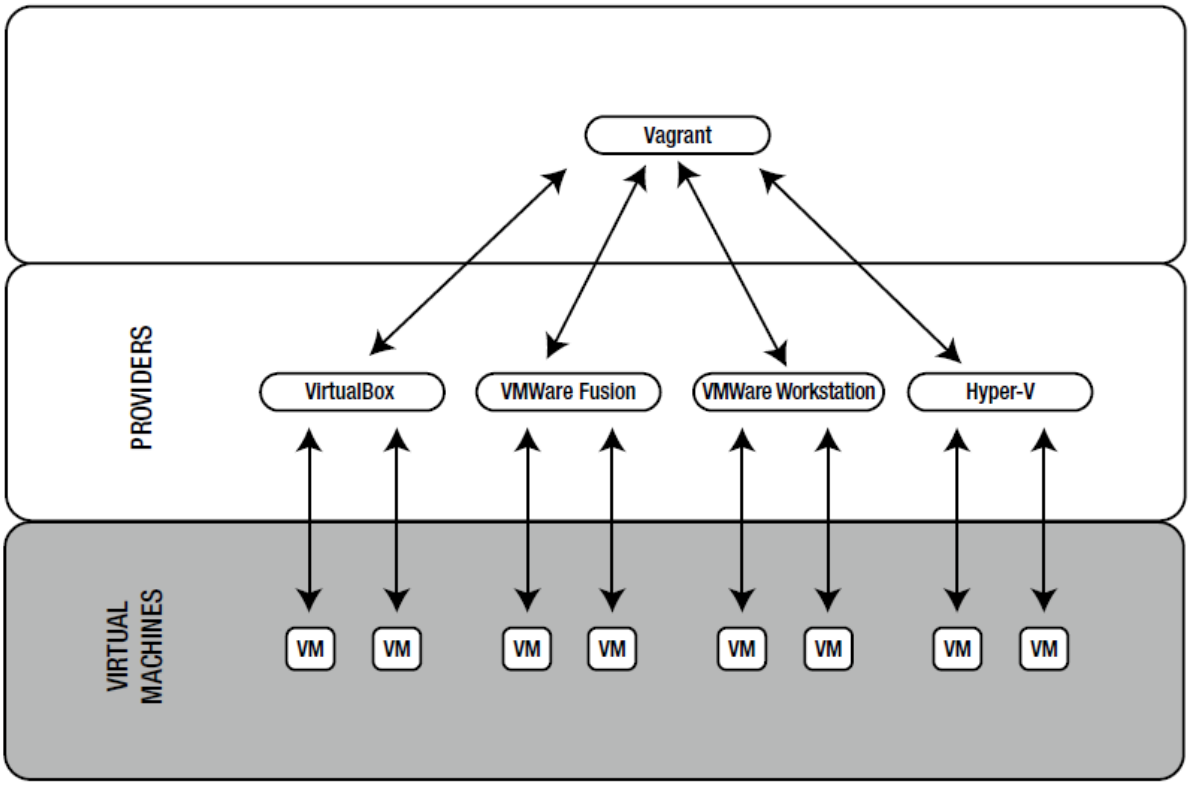
1. Download the VM image.
2. Start the VM.
3. Configure the VM's shared directories and network interfaces.
4. Maybe install some software within the VM.

With Vagrant, all these tasks (and many more) are automated. The command $ vagrant up can do the following (depending on the configuration file):

- Automatically download and install a VM, if necessary
- Boot the VM
- Configure various resources: RAM, CPUs, network connections, and shared folders
- Install additional software within the VM by using tools such as Puppet, Chef,
- Ansible, and Salt

Architrcture

Vagrant sits on top of existing and well-known virtualization solutions such as VirtualBox, VMWare Workstation, VMWare Fusion, and Hyper-V; and provides a unified and simple command-line interface to manage VMs. To work with Vagrant, you have to install at least one provider



In Vagrant terminology, a **provider** is a software virtualization solution such as VirtualBox, VMWare Fusion, or VMWare Workstation.

Commands

```
$ vagrant init [url]
$ vagrant up
$ vagrant halt
$ vagrant destroy [--force]
$ vagrant reload
$ vagrant ssh
$ vagrant status
```

Working

Set the Proxy on cmdline

```
set http_proxy=http://10.219.2.220:80
set https_proxy=http://10.219.2.220:80
```

Installation

1. Download the free VirtualBox for your operating system from the VirtualBox website.
2. After download, just run the binary and install it.
3. Download Vagrant.
4. Again, just run the binary to install it.

2.Vagrant is a command-line based tool. Once installation is complete, open a console window and create a new directory called 'vagrant_intro' to workwith new Vegrant baox

```
cd ~
mkdir vagrant_intro
cd vagrant_intro
```

3. To add a box, goto box repository: https://app.vagrantup.com/boxes/search Run this command:

```
$ vagrant box add <name>
$ vagrant box add ubuntu/trusty64
```

This will download the box named "hashicorp/precise64" from HashiCorp's Vagrant Cloud box catalog .

In the above command, you will notice that boxes are namespaced. Boxes are broken down into two parts - the **username and the box name** - separated by a slash

4.To create an environment, inside your folder run init command, it will create **'vagrantfile'**

```
vagrant init ubuntu/trusty64
```

It will downloads the Ubuntu Box into our local mechine, we can check the downloaded Box by going this location **Windows** : `C:\Users\<Username>\.vagrant.d\boxes` **Linux/Mac:** `~/.vagrant.d/boxes`

The generated 'Vagrantfile' is a Ruby file that controls your [one or more] virtual machines.

```
A 'Vagrantfile' has been placed in this directory. You are now ready to 'vagrant up' your
first virtual environment! Please read the comments in the Vagrantfile as well as
documentation on'vagrantup.com' for more information on using Vagrant.
```

5.Start the Environmet

```
$ vagrant up
```

6. connect to Environment

```
$ vagrant ssh
```

To Set Shared folder, edit vagrantfile as

```
config.vm.synced_folder "D:\\DevOps\\Instl\\VagrantBoxes\\SyncFolder", "/vagrant"
```

we named our syncd folder as "vagrant", you can find the files in Syncfolder by going /vagrant/

```
vagrant@vagrant-ubuntu-trusty-64:~$ cd /
vagrant@vagrant-ubuntu-trusty-64:/$ ls
bin   dev  home        lib    lost+found  mnt  proc  run   srv  tmp  vagrant  vmlinuz
boot  etc  initrd.img  lib64  media       opt  root  sbin  sys  usr  var
vagrant@vagrant-ubuntu-trusty-64:/$ cd vagrant/
vagrant@vagrant-ubuntu-trusty-64:/vagrant$ ls
chefdk_3.2.30-1_amd64.deb
```

Ubuntu Install

1.Install

```
sudo apt-get install vagrant
```

```
2.Complete log

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant$ mkdir centos

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant$ cd centos/

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$ vagrant box add centos/7

==> box: Loading metadata for box &apos;centos/7&apos;

    box: URL: https://vagrantcloud.com/centos/7

This box can work with multiple providers! The providers that it

can work with are listed below. Please review the list and choose

the provider you will be working with.


1) hyperv

2) libvirt

3) virtualbox

4) vmware_desktop


Enter your choice: 3
```

```
==> box: Adding box &apos;centos/7&apos; (v1804.02) for provider: virtualbox

    box: Downloading:
https://vagrantcloud.com/centos/boxes/7/versions/1804.02/providers/virtualbox.box

==> box: Successfully added box &apos;centos/7&apos; (v1804.02) for &apos;virtualbox&apos;!

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$ ls

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$ vagrant init centos/7A
`Vagrantfile` has been placed in this directory. You are now

ready to `vagrant up` your first virtual environment! Please read

the comments in the Vagrantfile as well as documentation on

`vagrantup.com` for more information on using Vagrant.

satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$ vagrant up

Bringing machine &apos;default&apos; up with &apos;virtualbox&apos; provider...

==> default: Importing base box &apos;centos/7&apos;...

==> default: Matching MAC address for NAT networking...

==> default: Checking if box &apos;centos/7&apos; is up to date...

==> default: Setting the name of the VM: centos_default_1537980466397_98016

==> default: Clearing any previously set network interfaces...

==> default: Preparing network interfaces based on configuration...

    default: Adapter 1: nat

==> default: Forwarding ports...

    default: 22 (guest) => 2222 (host) (adapter 1)

==> default: Booting VM...

==> default: Waiting for machine to boot. This may take a few minutes...

    default: SSH address: 127.0.0.1:2222

    default: SSH username: vagrant

    default: SSH auth method: private key

    default: Warning: Remote connection disconnect. Retrying...

    default:

    default: Vagrant insecure key detected. Vagrant will automatically replace

    default: this with a newly generated keypair for better security.
```

```
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it&apos;s present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: No guest additions were detected on the base box for this VM! Guest
    default: additions are required for forwarded ports, shared folders, host only
    default: networking, and more. If SSH fails on this machine, please install
    default: the guest additions and repackage the box to continue.
    default:
    default: This is not an error message; everything may continue to work properly,
    default: in which case you may ignore this message.
==> default: Rsyncing folder: /home/satya/Desktop/DevOps/vagrant/centos/ => /vagrant
satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$
satya@satya-Aspire-E5-523:~/Desktop/DevOps/vagrant/centos$ vagrant ssh
```

CentOS Default username/passwd is **root/vagrant**

How to Download Vagrant Box Manually

Save: box.sh, run on terminal

/* this is the box (and the version) that we want to download from:
https://app.vagrantup.com/debian/boxes/jessie64 */

```
wget https://app.vagrantup.com/debian/boxes/jessie64/versions/8.9.0/providers/virtualbox.box -
O debian-jessie64-8.9.0.box
```

/* add the box to vagrant */

```
vagrant box add debian/jessie64 debian-jessie64-8.9.0.box
```

/* update box version */

```
cd ~/.vagrant.d/boxes/debian-VAGRANTSLASH-jessie64/
```

mv 0 8.9.0

/* create metadata_url file */

```
echo -n "https://app.vagrantup.com/debian/boxes/jessie64" > metadata_url
```

/* show vagrant boxes */

```
vagrant box list
```

References

Installation : https://youtu.be/nZrOsxCPT2s

6. Microsoft Azure

Azure Scenario

Tisco, an IT and networking firm, develops, manufactures and sells networking hardwares to markets across globe. The organization is currently maintaining their own infrastructure but due to the massive growth of the organization in the global market, the organization is more keen towards increasing the manufacturing unit rather than expanding the existing infrastructure. Some of the challenges faced by the current infrastructure are as follows:

1. Exponential data growth is demanding more storage space resulting more maintenance cost.
2. Often resulting in application/database downtime while upgrading the server because of demanding resource.
3. 24*7 power supply and manpower is required hence adding to the cost.
4. Taking periodic backups of the critical servers and recovering it during fail over is complex.

In order to overcome the above challenges, Tisco has decided to gradually move some of their new deployments to Microsoft Azure.

Azure Introduction

Microsoft Azure is Microsoft's public cloud offering with a wide set of cloud services that gives the flexibility to build, manage and deploy infrastructure and application on a massive, global network using diverse technology and frameworks.

Azure service offerings fall into all the three categories:

- **Infrastructure as a Service (IaaS)**: IaaS offers virtualized servers and network infrastructure components so that users can easily provision and decommission as required
- **Platform as a Service (PaaS)**: PaaS offers resources/platform on which developers can build their own solutions.
- **Software as a Service (SaaS)**: SaaS offers complete software applications that are delivered as a cloud-based service. SaaS basically enables the users to easily access applications without the need to install them.

1.Creating Azure Account

1.Log on to Azure site and click on the "Start Free" button.

2. Create a Microsoft live id(For example, Outlook, hotmail, etc.)

3. Once you log in, fill in the required details

4. Access the Portal. Your portal is as follows.

5.To Launch Azure in powershell , run following commands

```
Install-Module Azure
Import-Module Azure
Add-AzureAccount
```

6.It will asks for email/password , once connect – it will show the Azure Account details

```
PS C:\windows\system32> Add-AzureAccount
Id                          Type Subscriptions                      Tenants
--                        ---- -------------                     -------
satyakaveti@outlook.com     User 197d6fbd-41b4-4                {bc2fc351-2b4d-}
PS C:\windows\system32>
```

7.To Know all subscription deatils , use - `Get-AzureSubscription`

```
PS C:\windows\system32> Get-AzureSubscription


SubscriptionId             : 197d6fbd-41b4-47fc-af46-95a94db6493d
SubscriptionName           : Free Trial
Environment                : AzureCloud
DefaultAccount             : satyakaveti@outlook.com
IsDefault                  : True
IsCurrent                  : True
TenantId                   : bc2fc351-2b4d-4678-9a94-7f395d617b51
CurrentStorageAccountName  :
```

8.To run powershell scripts we must ByPass the ExecutionPolicy as below

```
PS C:\windows\system32> Set-ExecutionPolicy -ExecutionPolicy Bypass

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution p
topic at http://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): a
PS C:\windows\system32>
```

2.Terminalalogies

The infrastructure of an application on Azure is made up of many components like virtual machine, storage account, web app, etc. Some of the terminologies used to referring these components are as follows

Resource: Resource in Azure are manageable items that are present in Azure, such as, virtual machines, networks, databases, web app etc

Resource groups : Resource group is a logical container that holds related resources for an entire application. While creating/deploying resources on Azure, you have to specify the resource group that has to be used for storing the resource.

Resource provider: Resource provider is a service that supplies the resources which you can deploy and manage through Resource Manager. For example, Microsoft.compute is one of the common resource provider which supplies virtual machine resource.

Some benefits of Resource Manager

- Deploy, manage, and monitor all the resources for a solution as a group
- Define the dependencies between resources to deploy them in the correct order.
- Organization's billing is transparent with the help of tags.
- Deploy solutions consistently throughout the development life cycle.

learn to create a resource group

1.Log in to the Azure portal.

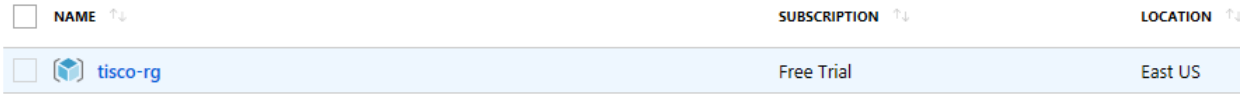
2.Navigate to the **"Resource group"** option towards the left pane.

3:Click the "Add" option in the top middle pane to create a new resource group.

Fill in the details as follows:

- Resource group name: **tisco-rg**
- Subscription: **Free trial**
- Resource group location: **East US**

You will observe that the resource group tisco-rg created as below

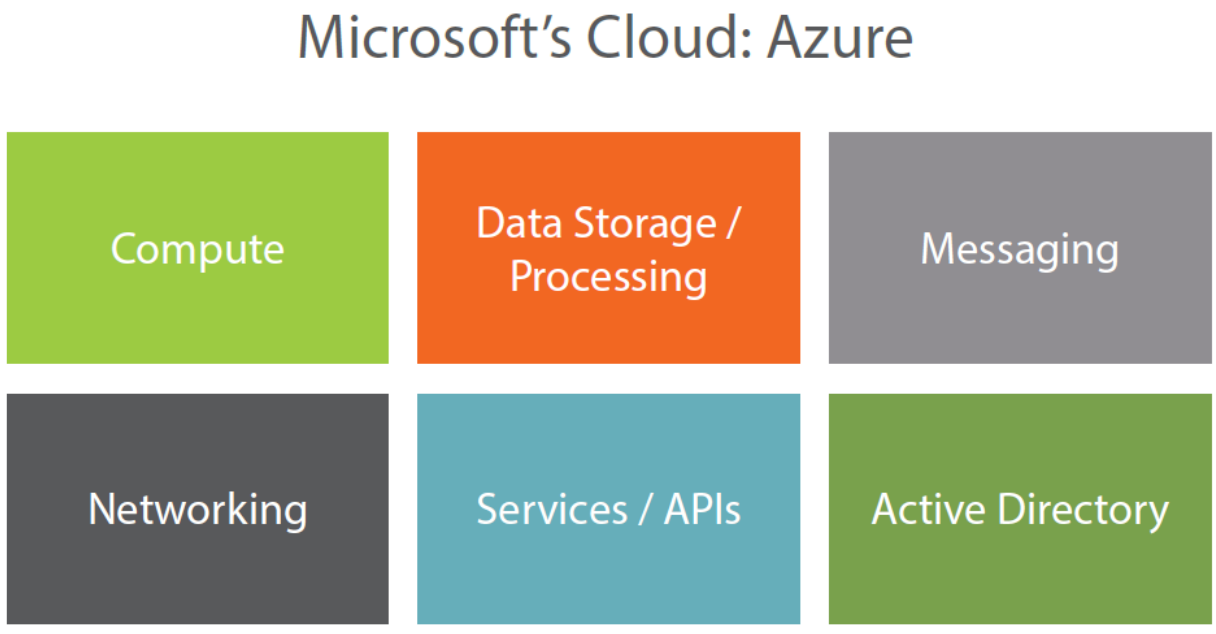


Tisco Project Raodmap

Tisco has decided to gradually migrate their existing infrastructure and also the new deployments to Microsoft Azure.

You will be performing the following tasks to achieve this

1. Create and Manage Storage
2. Configure and Manage Network
3. Provision and deploy Virtual Machines
4. Publish Web Applications
5. Manage SQL Database
6. Backup and Restore Data
7. Access Management using Active Directory



Azure Storage

Requirement : Tisco has huge amount of data stored in their file servers and in other object storages. As a part of their migration plan, Tisco wants to move the data to a storage which can accomodate terabytes of unstructured data.

Solution : Azure storage account can be used to store huge amount of data as files and also as objects.

Azure Storage provides data storage solution on cloud. It is highly scalable, elastic, globally accessible and automatically load-balances application-data based on traffic.

Azure Storage provides the following services:

- Table: Stores NoSQL data as key-attribute pair
- Blob: Stores unstructured object data on Azure like images, videos, documents
- Queue: Provides a reliable messaging solution for asynchronous communication between loosely coupled application components
- File: Offers file shares in the cloud using the standard Server Message Block (SMB) protocol

In order to ensure high availability and protect your data, the data must be replicated, either within the same data center or to another data center, depending on the replication option you choose.

Replication

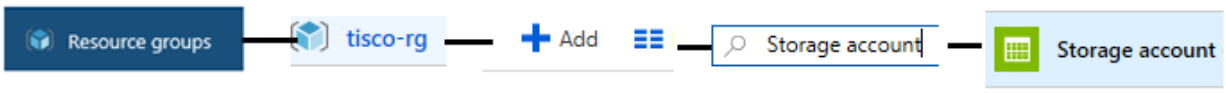
You can choose one of the following replication options:

- **Locally Redundant Storage (LRS):** Maintains three copies of data within the same data center in a single region
- **Geo Redundant Storage (GRS):** Maintains six copies of data, three copies in the primary region and three copies in the secondary region
- **Read-Access Geo Redundant Storage (RA GRS**): Similar to GRS. Only read permissions assigned to the data stored in secondary region

create a a general purpose storage account

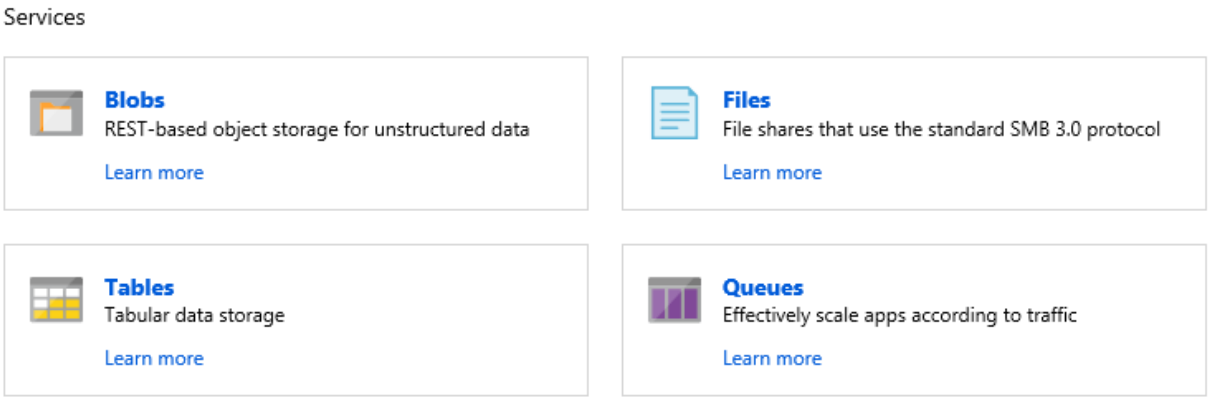
Step 1: Log in to the Azure portal.

Step 2: Navigate to the "Resource group" and open the tisco-rg and "Add" , search for "Storage account", click on it & create



Step 3: Fill in the details as follows and Click on "Create":

- Storage Account Name: **tiscostore**
- Location: **East US**
- Replication: **LRS**
- Resource group: **tisco-rg**



Azure Virtual Network (VNet)

Azure Virtual Network (VNet) is a representation of a network in the cloud. It enables Azure Virtual machines to communicate with each other, the internet and the on-premises network. Virtual Networks can be segmented into multiple Subnets

Azure Virtual Network provides the following key capabilities:

- **Isolation and segmentation**: Within each Azure subscription and Azure region multiple virtual networks can be configured. Each virtual network is isolated from other virtual networks.
- **Communicate with the internet:** By default, all resources in a virtual network can communicate outbound to the internet. For inbound communication with a resource, a public IP address(A public IP address is a resource with its own configurable settings) has to be assigned to it.
- **Communicate between Azure resources:** Azure resources communicate securely with each other.
- **Communicate with on-premises resources:** Azure VNet's can be integrated with on-premises resources.
- **Filter network traffic:** Inbound and out bound traffic through Azure VNet an be customized
- **Route network traffic:** Azure routes traffic between subnets, connected virtual networks, on-premises networks, and the Internet, by default. Traffic can be routed with azure route table or it can also be user defined

IP addresses can be assigned to Azure resources so as to communicate with other Azure resources, with the on-premises network and the internet. IP addresses can be of two types in Azure

IP Address Type	Definition	When to Use?
Public IP Address	A public IP address is an IP address that can be accessed over the internet	All public facing applications like Flipkart, Amazon etc., are assigned with a Public IP so that customers over the internet can access the application
Private IP Address	Private IP addresses allow Azure resources to communicate with others resources in a virtual network or an on-premises network	An SQL server hosted internally by an organization assigned with a private IP can only be accessed on-premises and is not exposed to the internet

Step 1: Login to Azure Portal.

Step 2: Create a new resource by navigating to "Create a resource".

Step 3: Provide the parameters for creating the Azure Virtual Network as follows.

```
Name: tisconet
Address space: 10.1.0.0/16
Resource group: tisco-rg
Location: East US
Subnet: default
```

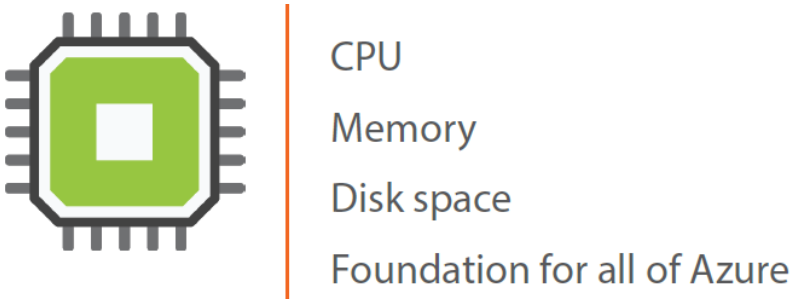
Step 4: You will observe that the Azure Virtual Network is created as below.

Azure Virtual Machines

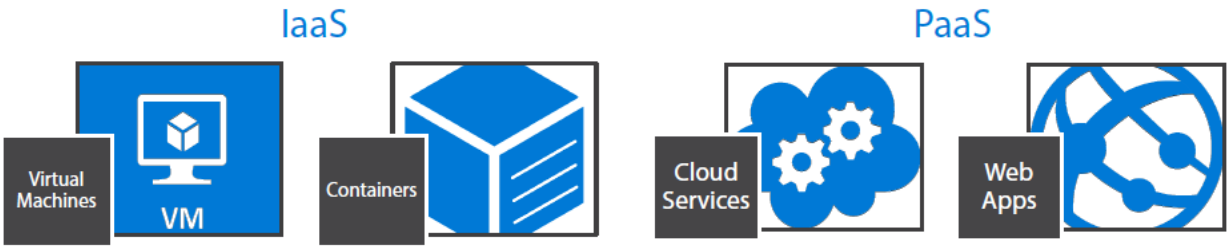
Azure Virtual Machines gives you the flexibility of virtualization without having to buy and maintain the physical hardware that runs it.

Azure Compute

Hardware / Resoures required to run our code



We have following Azure Compute Options



1.Virtual Machines

- Linux or Windows
- Prebuilt images
- Varying sizes
- Premium Storage
- You manage the operating system

2.Container

- Lightweight application hosts
- Chain images together
- Docker client support on Windows

3.Cloud Services

- Web / Worker Roles
- Package application code
- Declared target operating system
- Azure Service Fabric managed

4.web apps

- Web application code
- IIS hosting at scale
- Source control integration for CI
- Web Jobs for background processing

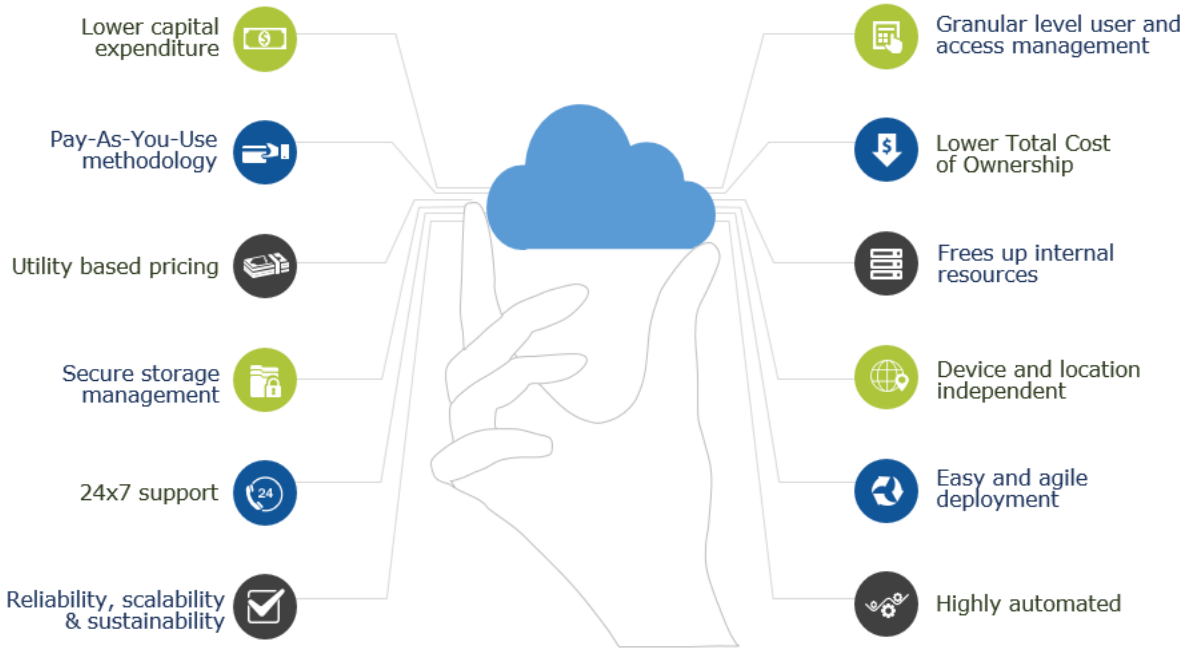
https://app.pluralsight.com/library/courses/chef-planning-installing/table-of-contents

https://app.pluralsight.com/library/courses/chef-planning-installing/table-of-contents

7. Amazon Web Services (AWS)

Cloud computing offers dynamic provisioning of resources based on demand, on a **pay-as-you-use** pricing. Instead of physical servers, cloud computing helps to spin out virtual servers. With dynamic scaling and load balancing features of cloud, long term planning is not necessary.

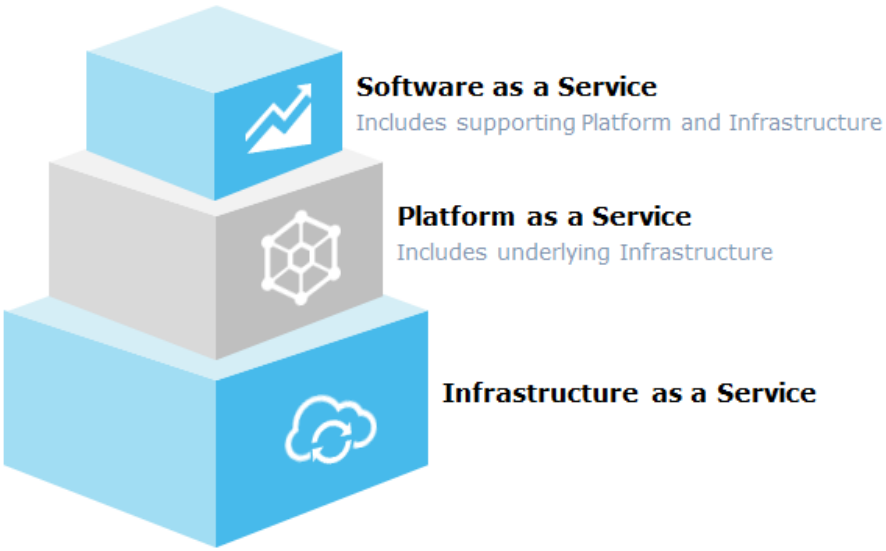
Why Cloud?



Amazon.com's Great Indian Festival or Flipkart's Big Billion Day, they also declare intermittent offer days.

Parameter	Normal business day	Promotional event day	Resulting business concern
Web traffic	0.3M requests	1.5M requests	Business continuity without affecting end user experience because of scalability issues
No. of servers	250	5000	
Staff required	7000	19000	Management and cost issues
Staffing and management cost	7.25M INR	32M INR	Huge and indeterminate capital investment required
Maximum network spike	0.7 MB/sec	7.3 MB/sec	Improper infrastructure usage leading to servers being over or under utilized
Failure rate	3%	112%	Loss of trust in business
Data volume generated	10 GB	43 TB	Data loss or corruption concerns

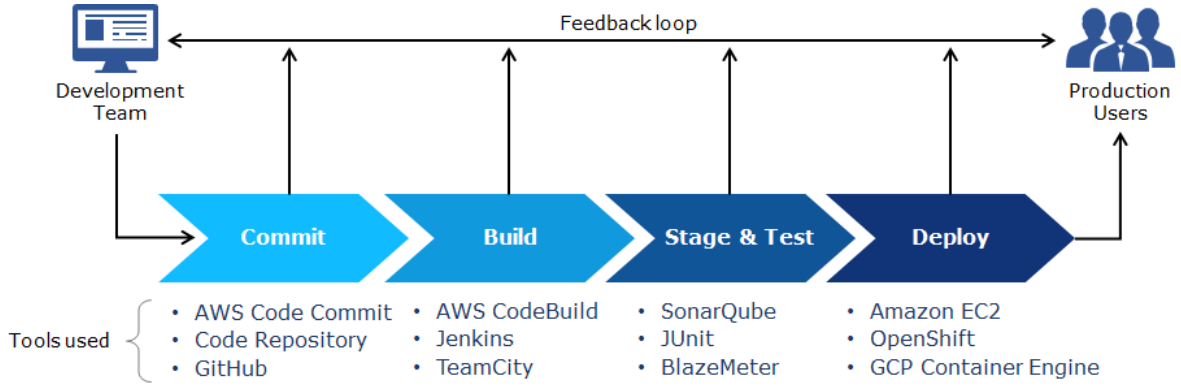
Business concerns	Issues faced in data center based model		Cloud solution
	Observation	**Inference**	
Scalability issues	• Team needed a month's time to mobilize infrastructure for the event	Limited scalability and high turn-around time	**Highly scalable with very minimum turnaround time** Quick and limitless scaling
Reliability and quality concerns	• Traffic spikes, huge load disrupted measures like performance, quality • Systems did not respond	Measures hard to achieve	**Easy to achieve measures** Automated and easily attainable Performance, quality and reliability
Indeterminate cost concerns	• Common budget out-runs • Cost was 4 fold during festive seasons compared to a normal business day	Less cost effective	**Highly cost effective** Setup and maintenance is optimally mapped to budget restrictions
Management overheads	• Resources needs continuous monitoring, maintenance and management • Company spent 15 million for staffing and maintenance	High management and maintenance overheads	**No maintenance and easy monitoring** Maintenance done by CSP Monitoring is easy through a front end dashboard
Utilization of resources	• Resources either overused or underused • Could mobilize only 5000 servers when 6500 needed at peak load	Poor utilization of resources	**Maximum utilization of resources** On-demand resource provisioning helps in avoiding over/under utilization
Infrastructure elasticity	• Resources worth 17 million which was invested remain unused • Manual addition and disposal of resources needed dedicated staffing	Highly non-elastic	**Highly elastic** Dedicated staff not required due to automated addition and removal of resources
Set-up and access concerns	• Data lost or corrupted when load was high • At peak, 3GB worth of data was lost or corrupted	Access management and setup is difficult	**Easy setup and access grants** Resource setup and access rules can be provided with few clicks



Deployment model	Best suited for	Advantages	Challenges
Public cloud	• Variable workload • Test & Dev	• Lowest TCO • Rapid elasticity • Faster deployment	• Data Security • Privacy
Private cloud	• Sensitive Data • Legal compliance	• Security and control • Optimized performance	• High cost of ownership • Skillset
Hybrid cloud	• Cloud bursting • On demand access • Sensitive data	• Lower TCO • Rapid elasticity • High Performance • Highly customizable	• Portability • Migration • Integration
Community cloud	• Collaboration of universities, Groups of hospitals	• Lower TCO than private cloud • Rapid elasticity	• Complex IT governance • Skill set

General Information	Cloud service providers		
	AWS	**Microsoft Azure**	**Google Cloud Platform**
Year of Launch	2006	2010	2012
Service models adopted	IaaS, PaaS, SaaS	IaaS, PaaS, SaaS	IaaS, PaaS, SaaS
Market share (%)	47	10	4
Global infrastructure distribution	49 availability zones within 18 geographical regions	36 distinct regions	5 Global regions and multiple zones within it
Billing pattern	Pay-per-use, Per-second, Per GB hours	Pay-per-use, Per-second, Per GB hours	Pay-per-use, Per-second, Per GB hours
Top customers	NASA, Netflix, CIA, Uniliver, Kellogs, Adobe, BMW, Siemens, Nokia, SAP, Vodafone	DishTv, Mosaic, Frame, University of Victoria, Brain-shark	Apple, Coca-Cola, Whirlpool, Airbus, Philips

Continuous Delivery and Cloud



- AWS - Compute and Network Services
- AWS - Storage and Database Services
- AWS – Cloud management Systems
- GCP - Storage and Database Services
- GCP - Compute and Network Services

1. History

Challenges:

- Faced major database corruption in 2008
- Storage crunch for ever increasing volumes of data
- Extreme difficult in scaling within their data centers
- Unpredictable rise in streaming hours of TV shows

NETFLIX

100 million members

190 countries

Benefits after using AWS:

- Quickly deploy thousands of servers within minutes
- Cost benefits
- Scalability
- Increase resiliency and efficiency
- Improved customer experience with real time network monitoring

125 million hours of TV shows and movies

Challenge:

- User interface should scale automatically across platforms and various interface
- Search interface should allow user to find what they looking for, view from gallery etc.
- Ability to download the images with ease and share it on Facebook, Twitter or Google+ or Pinterest etc.
- An Application Programming Interface (API) for automated uploads, authentication and authorization

Benefits after using AWS:

- Easy access to the wonders of the space
- Delivers a treasure of pictures, videos and audio files from one centralized location
- On demand and scalability
- Agile in business

Public access to over **140,000** images, audio recordings and videos etc.

https://youtu.be/jOhbTAU4OPI

Amazon Web Services(AWS) is a low cost cloud service platform from Amazon, which provides services such as compute, storage, networking, CDN services etc to users. All AWS services are exposed as web services accessible from any where, any time, on a pay per use pricing model.

AWS services can be managed through a web based management console, command line interface (CLI) or software development kits (SDK). With AWS, you can provision resources in seconds and build applications without upfront capital investment.

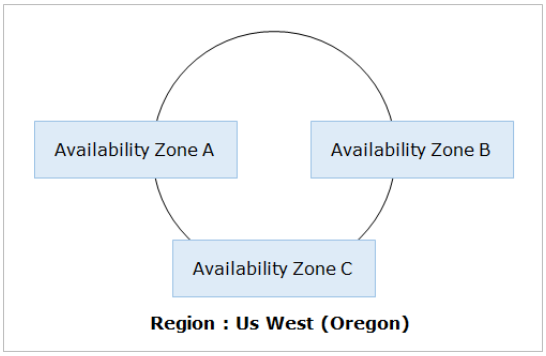
Choosing Region

Region

- is a physical location spread across globe to host your data
- In each region, there will be atleast two availability zones for fault tolerance
- Regions are completely separate from one another
- Enterprises can choose to have their data in a specific region

Availability zones

- Availability zones are anologous to clusters of datacenters
- Availability zones are connected through redundant low-latency links
- These AZs offer scalable, fault tolerant and highly-available architecture



Most of the AWS services are region dependent and only a few are region independent. Few services may not be available in all the regions. So, while determining a region to push the workloads, the following parameters to be considered.

- Availability of required services
- Cost
- Latency
- Security&Compliance
- Service Level Agreements(SLAs)

PluralSight Started : https://app.pluralsight.com/library/courses/aws-developer-getting-started

2.Services

55 services currently available from AWS.The following are the various categories of services offered by Amazon Web Services (AWS).

Services	Description
Compute	• To provision Virtual compute machines • Deploy applications on elastic beanstalk • Scheduling jobs and triggering actions based on events using AWS Lambda
Storage	• Storing frequently accessed and infrequently accessed data • Archival storage options
Database	• Relational database such as MySQL • No SQL database such as Dynamo DB • Building a Data warehouse
Networking and Content Delivery	• Creating a Virtual private network • Creating and Managing domains using Route 53 • Delivering content using Direct Connect

Compute
EC2
Lightsail
Elastic Container Service
EKS
Lambda
Batch
Elastic Beanstalk

Storage
S3
EFS
Glacier
Storage Gateway

Database
RDS
DynamoDB
ElastiCache
Neptune
Amazon Redshift

Developer Tools
CodeStar
CodeCommit
CodeBuild
CodeDeploy
CodePipeline
Cloud9
X-Ray

Management Tools
CloudWatch
AWS Auto Scaling
CloudFormation
CloudTrail
Config
OpsWorks
Service Catalog
Systems Manager
Trusted Advisor
Managed Services

Networking & Content Delivery
VPC
CloudFront
Route 53
API Gateway
Direct Connect

All Services provided by AWS

Security, Identity & Compliance
IAM
Cognito
Secrets Manager
GuardDuty
Inspector
Amazon Macie
AWS Single Sign-On
Certificate Manager
CloudHSM
Directory Service
WAF & Shield

Internet of Things
IoT Core
IoT 1-Click
IoT Device Management

Customer Engagement
Amazon Connect
Pinpoint
Simple Email Service

Machine Learning
Amazon SageMaker
Amazon Comprehend
AWS DeepLens

Analytics
Athena
EMR
CloudSearch
Elasticsearch Service

Local Development	AWS Development
Everything runs locally on same machine	One Service = One Responsibility
Files stored on server	Files served from S3
Database on same server as application	Database as a Service
Connections done manually	Connections done with AWS SDK

AWS CLI: AWS Command Line Interface

AWS CLI: Made for Operations Engineers morethan Developers

- Great for Shell Scripting
- Feature Complete with Console & SDKs
- Interact with Any Service

AWS SDK: Software Development Kits

AWS provides SDK's for all programming languages for implementing application inside AWS. For example the AWS SDK for Java is a collection of tools for developers creating Java-based Web apps to run on Amazon cloud components such as Amazon Simple Storage Service (S3), Amazon Elastic Compute Cloud (EC2) and Amazon SimpleDB

Some of the AWS SDK api's are

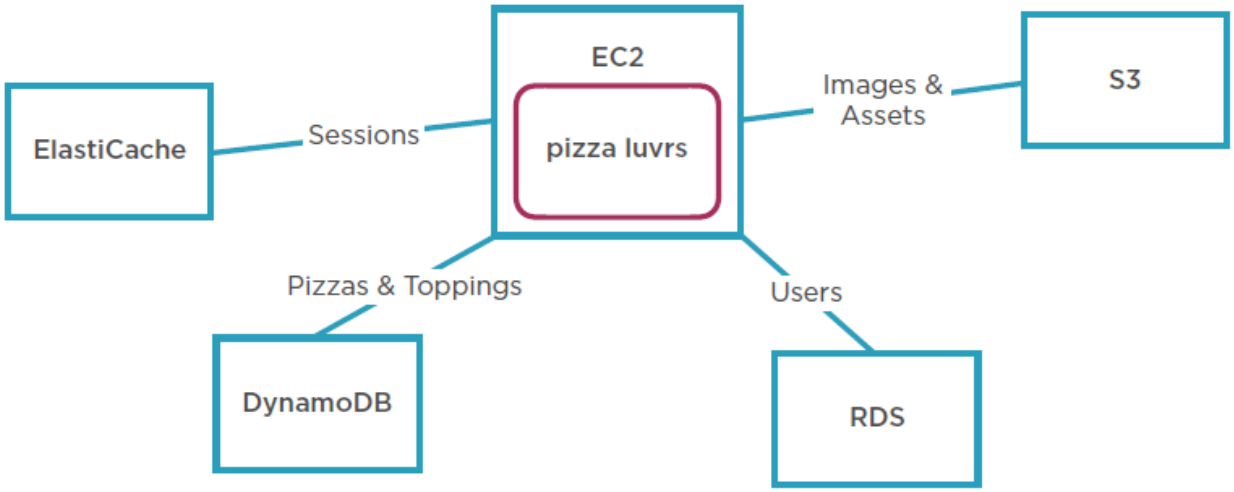
- Java
- .Net
- Node.js., etc

AWS Access Key gives access for SDK & CLI

Example : Pizza Application

Local System:

- Nodejs
- Application Hosting > **EC2**
- Images/Assets in website > **S3**
- User Registrtions Database > **RDS**
- User Sessions Storage Cache > **ElastiCache**
- Saving Pizza's NoSQL Db > **DynamoDB**



Installing AWS Commandline Interface

1. Go to https://aws.amazon.com/cli/ follow the installation steps on right side menu, in windows it is

> **Windows**
> **Download and run the 64-bit or 32-bitWindows installer.**

2.After installation check by running `>aws -version` command on cmd

```
C:\Users\kaveti_S>aws --version
aws-cli/1.16.17 Python/3.6.0 Windows/10 botocore/1.12.7
```

3. **Creating and Initializing an AWS Account**

- go to https://aws.amazon.com/ & **Sign into Console**

4. **An AWS Access Key is required for configure the CLI in local system& SDK**

- Top menu > Your name > Security Credencials >Continue to Security Credencials
- Expand AccessKeys > Create New Access Key > Copy & Download file(rootkey.csv)

```
Access Key ID         :AKIAJJAJRA4MN3H62CCA
Secret Access Key     :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
```

5.**Configure CLI In Local System**

- Type "`aws configure`" on command prompt & provide required details

```
C:\Users\kaveti_S>aws configure
AWS Access Key ID [None]: AKIAJJAJRA4MN3H62CCA
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-2
Default output format [None]: json
```

- To test the configuration use, `aws ec2 describe-instances`

```
C:\Users\kaveti_S>aws ec2 describe-instances
{
    "Reservations": []
}
```

3. CloudWatch & IAM

CloudWatch

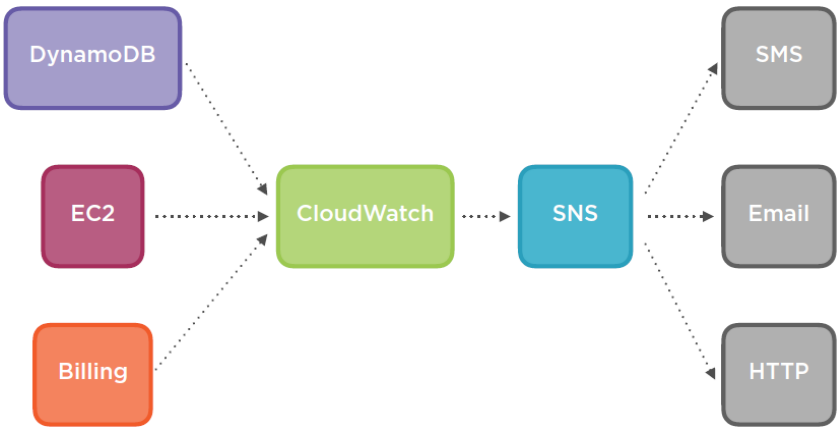
Set Alarms & Notifications in response to Events

IAM

Simple User and Access Management

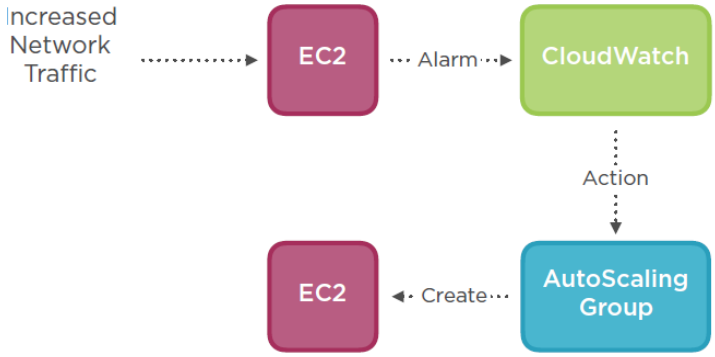
1.CloudWatch

Cloudwatch is a **Service to set alarms based** on Service metric thresholds. That means send a notification when a perticlar event is occurred using **Simple Notification Service(SNS).**



Examples of CloudWatch Alarm Actions

- Send Notification via SNS
- Trigger AutoScaling Action
- Trigger EC2 Action



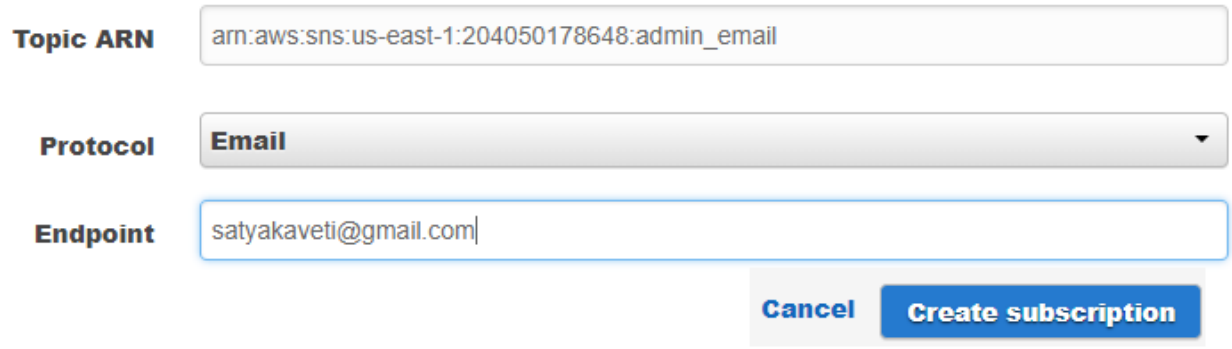
Task 1 : Configuring Simple Notification Service

1. Set Region as **US East (N. vergina)**

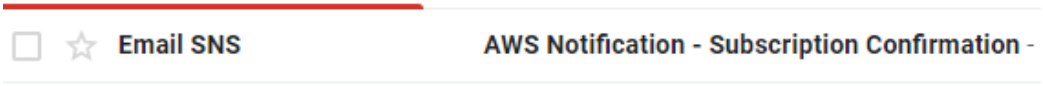
2.Go to services, select **SimpleNotificationService(SNS) > Getting Strarted**

3. Create New Topic > Topic Name: **admin_email** ; Display Name **: Email SNS**

4.Subscriptions > Create Subscription > Protocal : **Email**, EndPoint: **myemail@gmail.com**



5.A confirmation mail is sen to your mail id. Check & Confirm subscription.

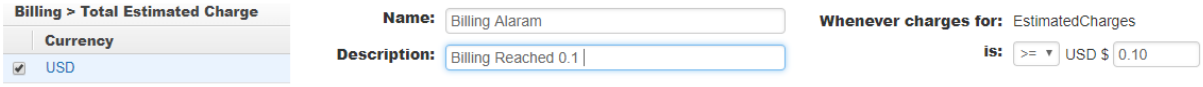


Task 2 : Creating a CloudWatch Alarm : "Billing Alaram"

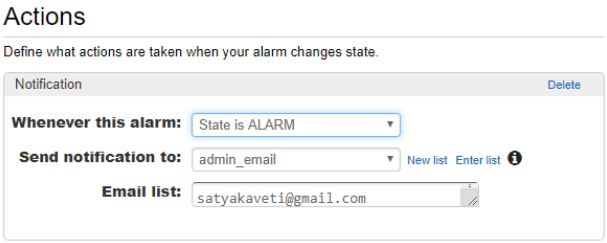
1.Go To Top Menu > Your name > My Billing DashBorad > Preferences > Check[] box for

2.Go to Services > Cloudwatch > Left Menu : Alarams > Create Alaram

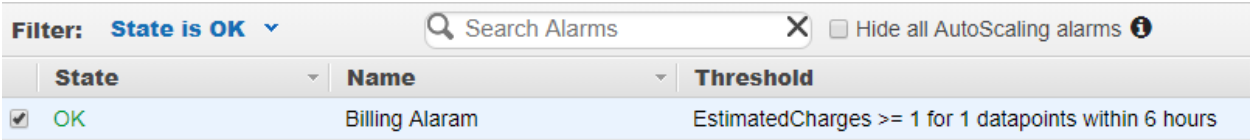
- **Select metric** > Total Estimated Charge > Check USD >Set Alaram Threshold



- **Set Actions & create Alaram**



3.Refresh it, Billing alaram is created



2.IAM - Identity & Access Management

Identity & Access Management (IAM) Service to configure authentication and access for user accounts

By using IAM, we can manage

- Passwords
- Multi-Factor Authentication
- Access Keys
- SSH Keys

Multi-Factor Authentication(MFA)

Authentication that requires more than one factor to authenticate

Task 1 : Securing Your AWS Account with MFA

1.Go to Top > Your name > My Security Credencials > Select : Multi-factor Authentication > Activate

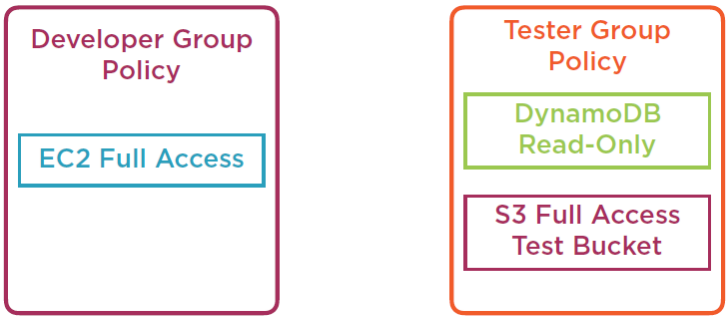
2.Select [.] Virtual device(Your Smartphone) & install **AWS MFA-compatible** app on the smartphone. So Install MFA apps like Google Authenticator, open it.

3.Scan the QR code using the App, add the two Conseqgutiive keys from app & Finish

4.Signout from aws, try to re-login, it will asks for MFA code. That's it!!

IAM Policy

Used to manage perimissions for different groups



Root Account Permissions

- Administer all services
- Modify billing information
- Manage users

IAM Policy Statement Properties

- **Effect** : "Allow"or"Deny"
- **Action**: Operation user can perform on services
- **Resource** : Specific Resources user can perform action on

To check go to My Security Credencials > Policies > Check AdministratorAccess

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

1.**To Create user**, Go to Servies > IAM >Add User > username:[ ] > Create User

2.To Add user to Group, Add new Group> Choose Group name > Add

3.It will create Access Keys for user, go to command line add user keys to system

```
C:\Users\kaveti_S>aws configure
AWS Access Key ID [****************2CCA]: AKIAJFAF2CG3QFRQNP3A
AWS Secret Access Key [****************M1S7]: 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
Default region name [us-east-2]: us-east-2
Default output format [json]: json
C:\Users\kaveti_S>aws ec2 describe-instances
{
    "Reservations": []
}
```

Password Policy

- Require at least one uppercase letter
- Require at least one lowercase letter
- Require at least one number
- Require at least one non-alphanumeric character
- Allow users to change their own password

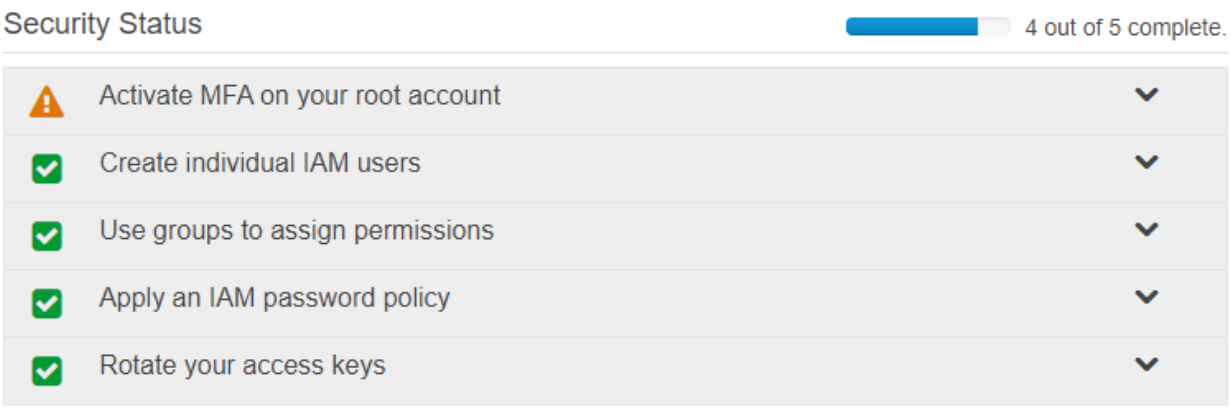
To set these options Go to My Security > Account Settings > Password Policy

Task 2 : Add password to Creted user

Iam > Users > user : smlcodes > Security credentials Tab > Console password : **Manage**

IAM users sign-in link:**https://204050178648.signin.aws.amazon.com/console**

Finally we completed with Security Things



4. EC2 & Virtual Mechines

Amazon **EC2(Elastic Compute Cloud**) provides users with the means of creating instances. These instances are virtual machines that are created on the AWS infrastructure.

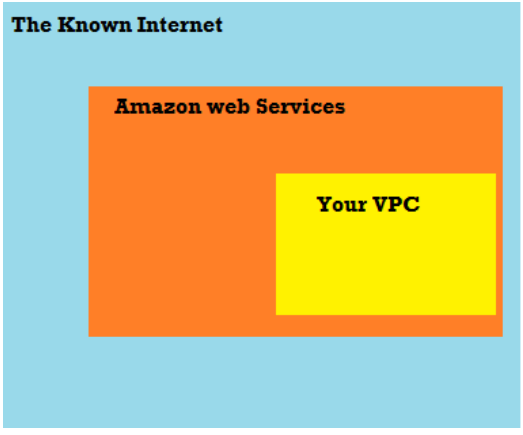
The EC2 provides users with the opportunity to choose among several varieties of operating systems, RAMs and CPUs.

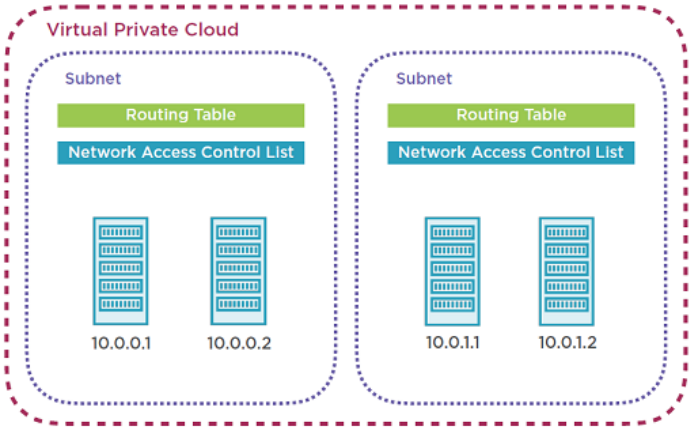
You can imagine EC2 as a computer which is maintained by someone else.

1.Virtual Private Cloud

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined.

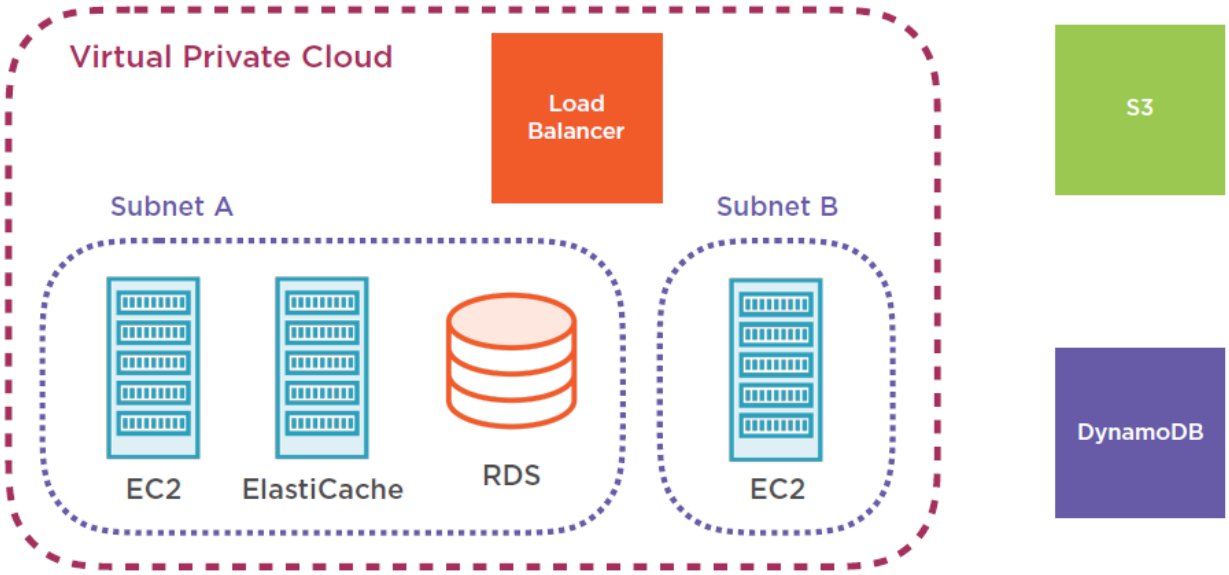
A "Virtual Private Cloud" is a sub-cloud inside the AWS public cloud. Sub-cloud means it is inside an isolated logical network. Other servers can't see instances that are inside a VPC. It is like a vlan AWS infrastructure inside a vlan.





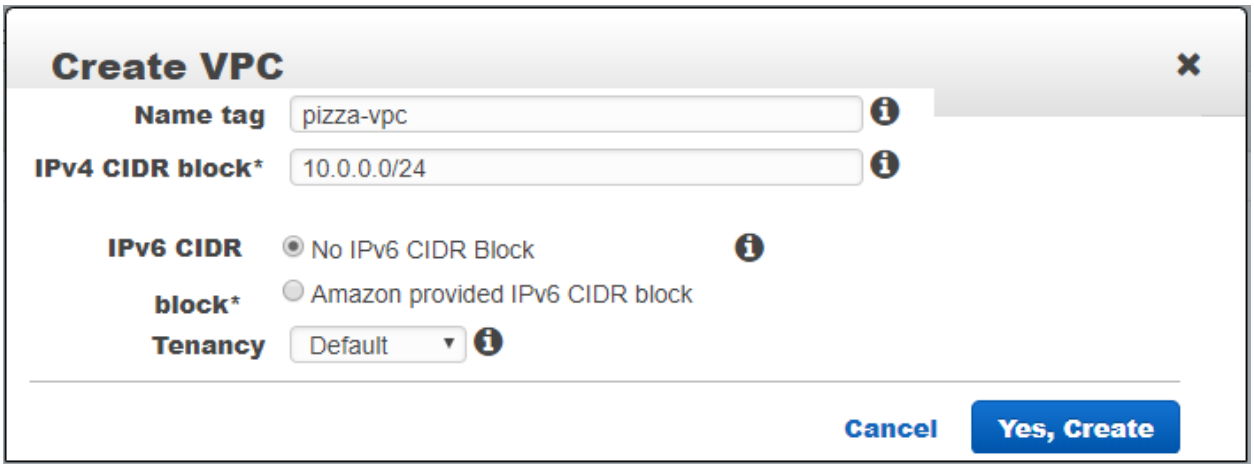
Task : Creating a Virtual Private Cloud

We are now going to create below archetecutre in AWS



1.Go to Sevices > Search > VPC

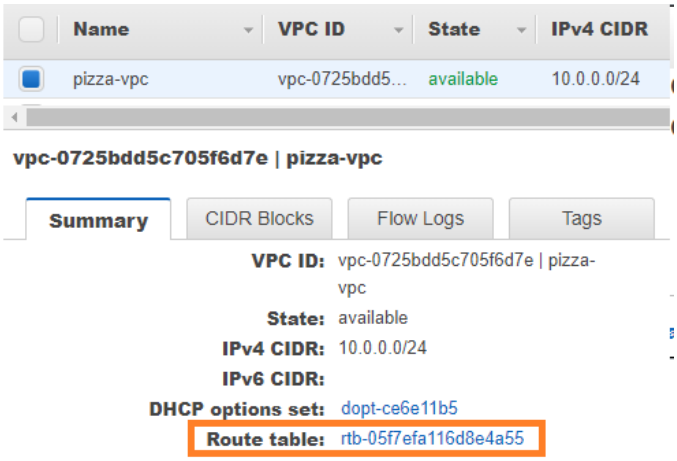
2.Click on VPC > Create VPC > fill details & Create



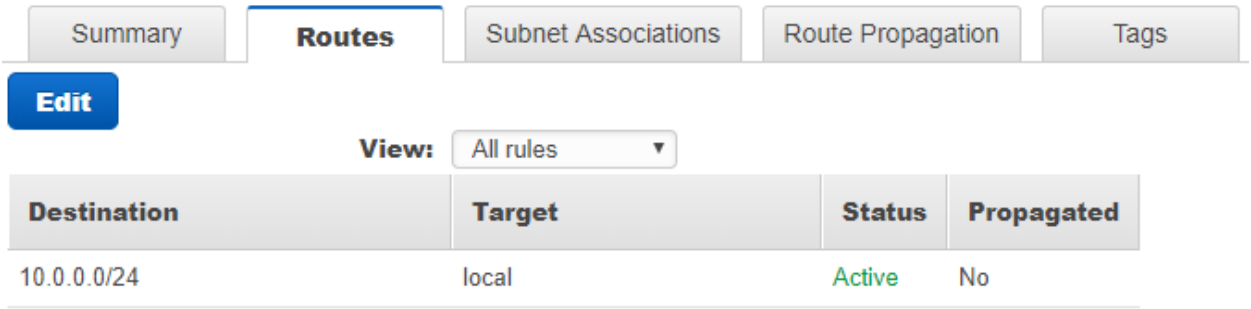
3. VPC is created Successfully.

Now VPC is just created, but for instances of the VPC has no access with Internet by default. We need to configure Routing table for the Internet access.

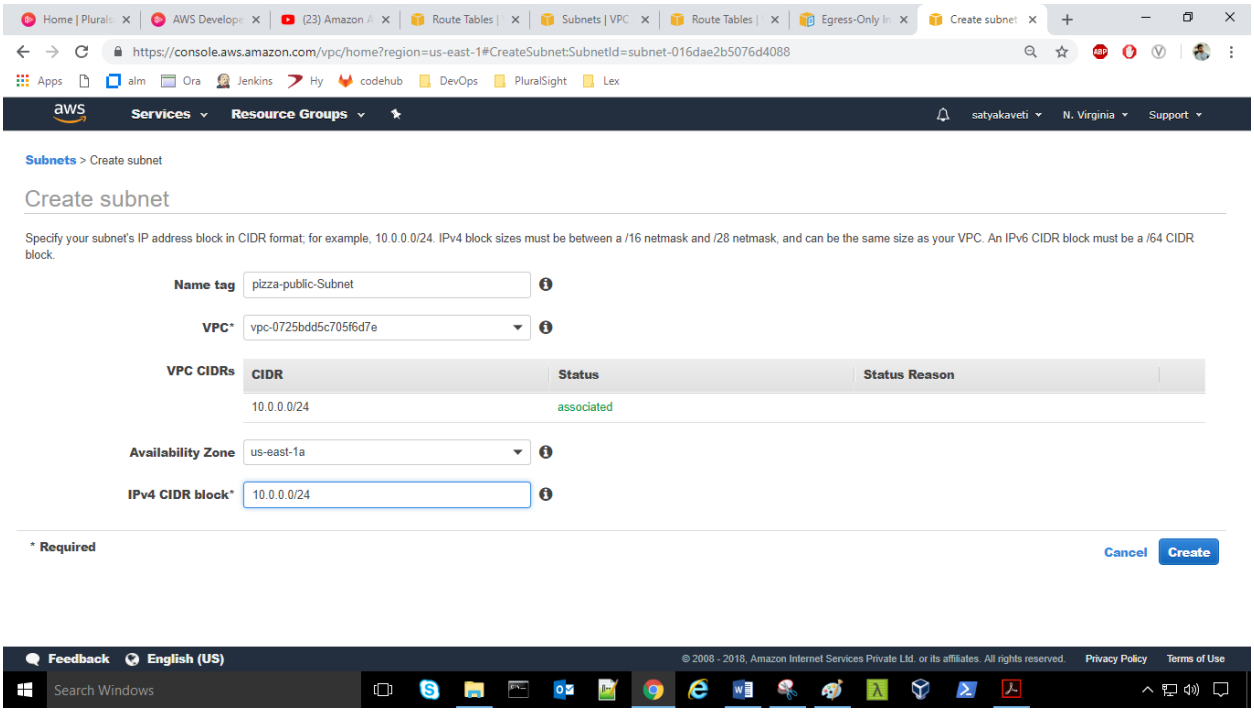
4.**To Configure VPC,** Go to VPC > Left: Your VPC's > Select : Pizza VPC > Summary: Routetable



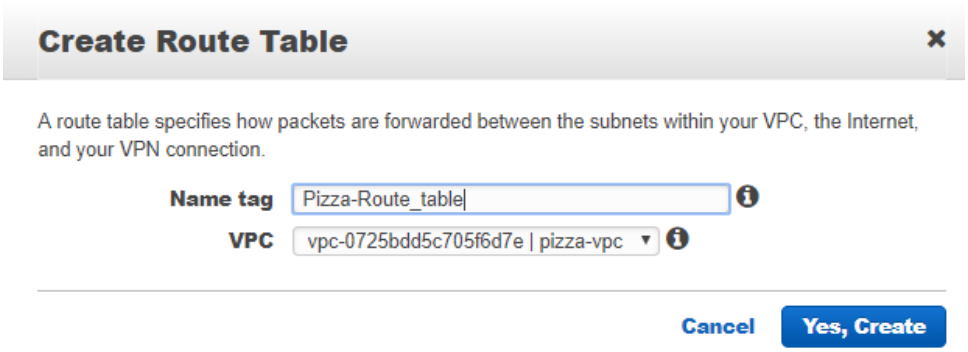
By Clicking Route Table link, it will open new tab, and select Route Table Id > Routes Tab



5.**Next, Create Subnet**, VPN >Left: Subnet > Create Subnet

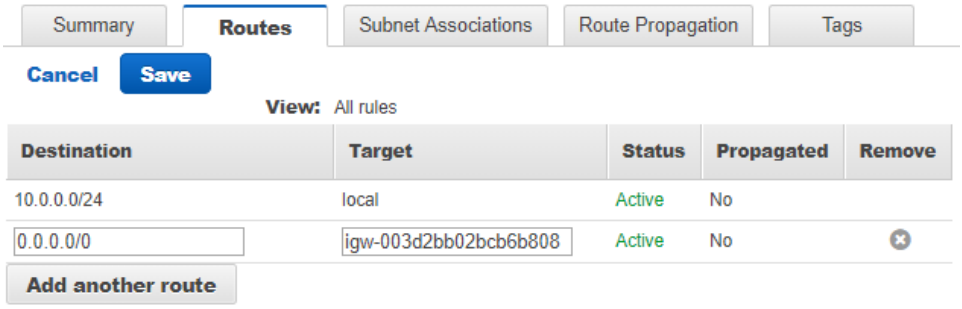


6.**Next, create Route Table**, VPN >Left: Route Table > Create Route Table



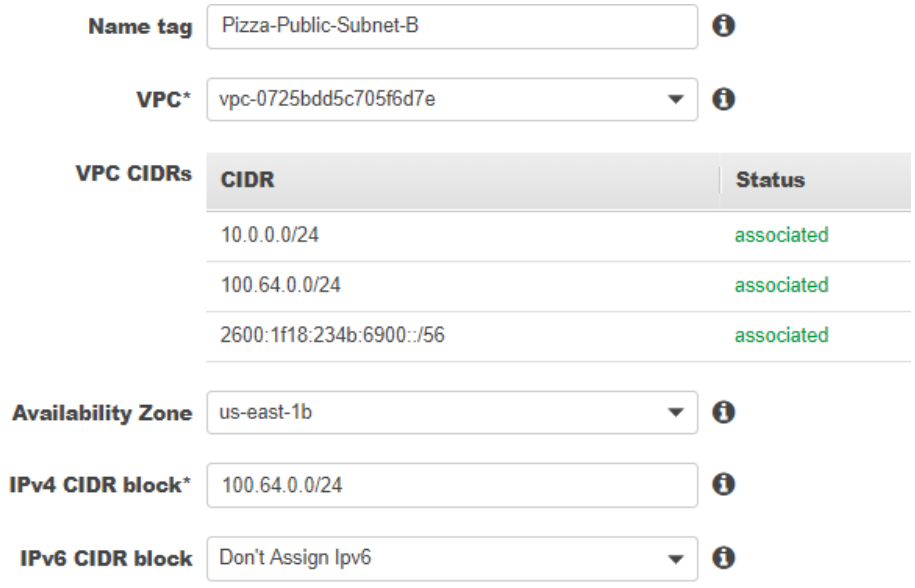
7.To access Internet, we must create Internet gateway : **Internet gateways** > Create internet gateway

8.To add Internet gateway to VPC, go to subnets > Pizza-Public-Subnet>Routing table > Edit > add another route & save details



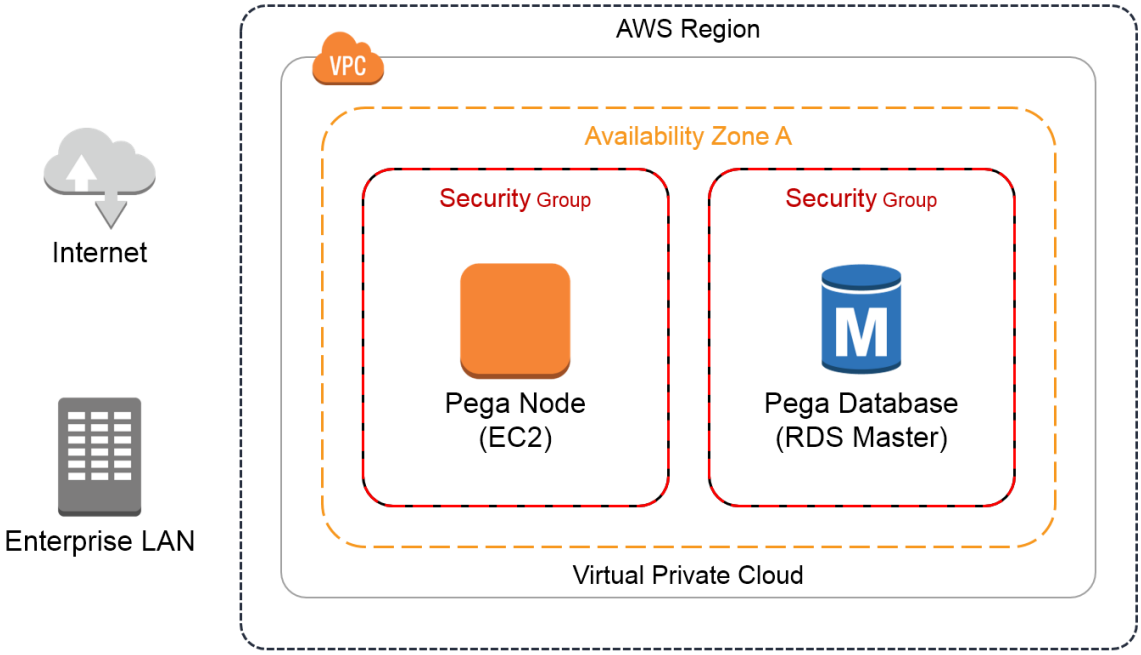
8.So now we need to create another subnet in different region for replica purpose.

- Go to **VPCs** > Edit CIDRs > **Add IPv4 CIDR > 100.64.0.0/24 > save**
- Go to VPC > **Subnets** > Create subnet > Fill Details > Save



2.Elastic Cloud Compute (EC2)

EC2 is a webservice that enables you to launch and manage Linux/Unix/Windows operating system instances in Amazon data centers.



1.EC2 Instance Parameters

- CPU
- Memory
- Storage
- Network

2.EC2 popular OS's

- Linux (Amazon, Red Hat, Ubuntu, etc)
- Windows

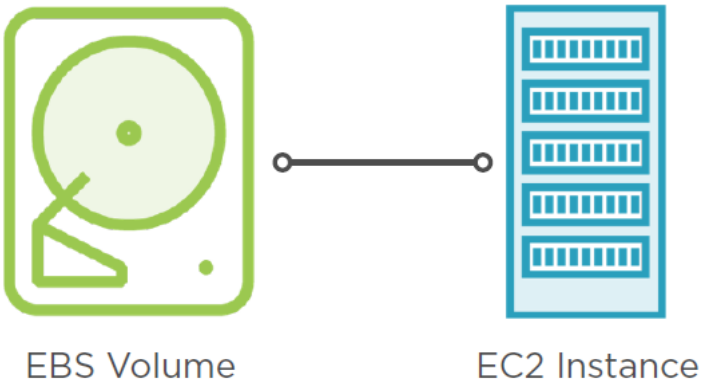
3. Amazon Machine Image (AMI)

- Operating System which is installed on EC2 is called as **AMI**
- Operating System + Software installed on EC2 instance
- Examples of AMI are
    - Anti-Virus Scanners Network Firewall
    - Business Intelligence Software

EC2 requires Storage space for Uploding & Saving files. For that they provide default service as **"Elastic Block Store"**

4.**Elastic Block Store**

- Independent storage volumes used with EC2 instances

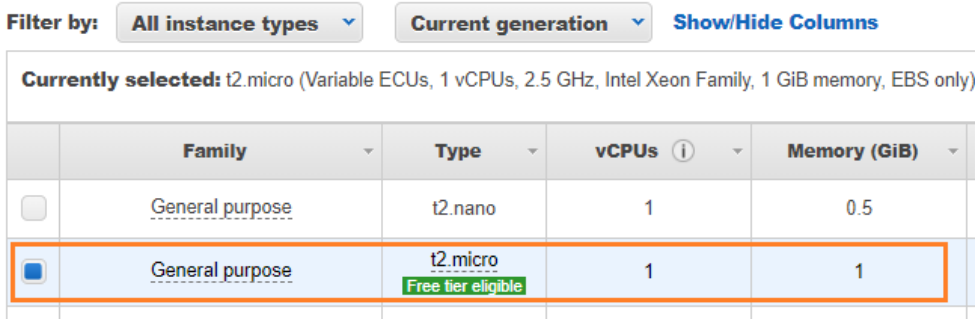


Task : Creating an EC2 Instance

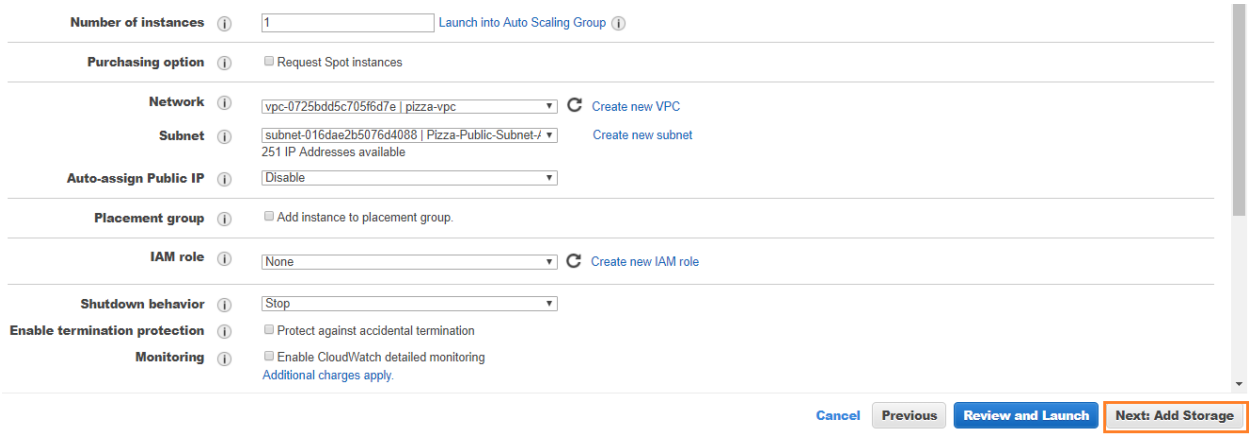
Go to Services > EC2 > Create Instance >Launch Instance.

1: Choose an Amazon Machine Image (AMI)

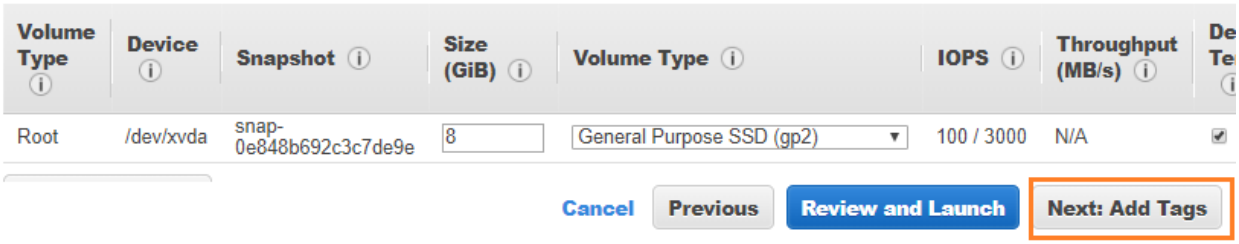
2: Choose an Instance Type



3: Configure Instance Details



4: Add Storage



5: Configure Security Group & Launch

All Traffic

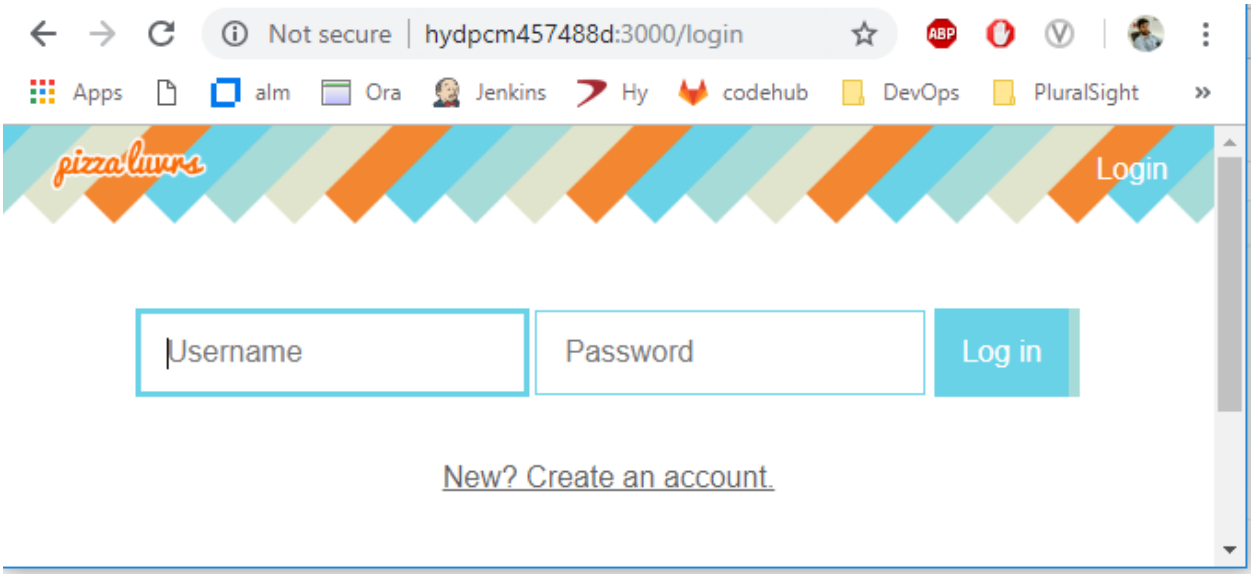
6: Important !! Download Key pairs

Task 2 : Connecting to an EC2 Instance & Deploying our Application

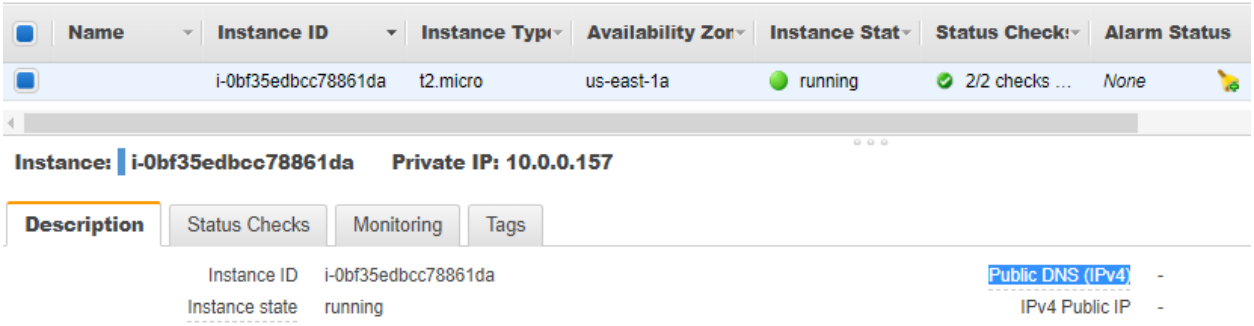
1.**Download** Our Application code , extract it , navigate to the extracted location from cmdline

2.Install npm packages & Start npm

```
D:\DevOps\Instl\aws\PizzaApp>npm install
D:\DevOps\Instl\aws\PizzaApp>npm start

> pizza-luvrs@1.0.0 start D:\DevOps\Instl\aws\PizzaApp
> node index.js

(node:640) [DEP0022] DeprecationWarning: os.tmpDir() is
Server running at:  http://HYDPCM457488D:3000
```



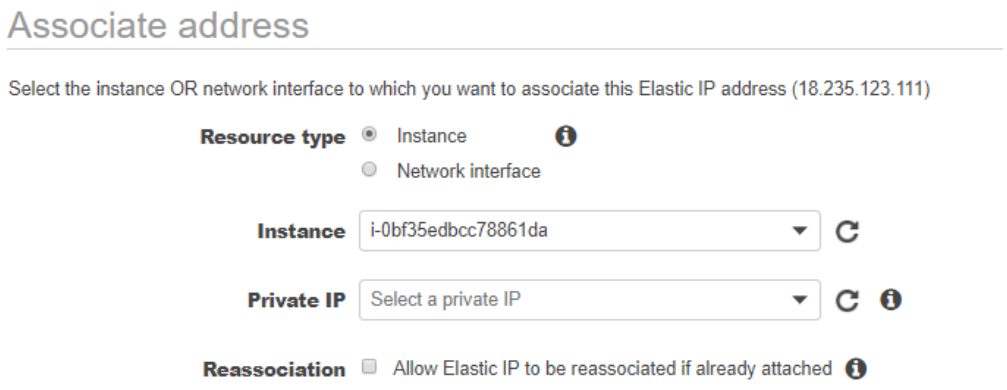
3.Go to EC2 > Running Instances > Click : 1 Running Instances, see there is **no Public DNS (IPv4)**



4.To connect with our EC2 instance form our local machine, we need to configuere Public DNS**(Public IP Address).**

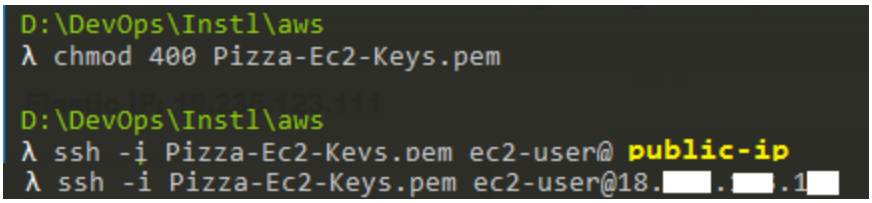
5.**Elastic IP** instace will manages the Public IP addresses that are creation, destroyed, and assigned.

- EC2 > Left: NETWORK & SECURITY > Elastic IPs > **Allocate new address > Create**
- Now, Select Elastic IP > Actions > Associate Address > Click Associate



- If you check the Public ip Address in running instances, it is not empty !

Task 3 : Connect to the EC2 Instance via SSH



Task 4 : Connect to the EC2 Instance via Putty from Windows

1.Open PuttyGen > Load > Select Perm key > Save Private Key

2. Open Putty, Host : Enter Public DNS (IPv4)

3.Connection > SSH > Auth > Locate Prieve key

Task 5 : Installing PizzaApp

1. Installing node.js on Instance

```
sudo yem update
curl –location https://rpm.nodesource.com/setup_6.x | sudo bash -
sudo yum -y install nodejs
node -v
```

2.Using WinScp, move PizzaApp code to /PizzaApp folder & run following commands

```
npm install
npm start
```

3. Scalling Application

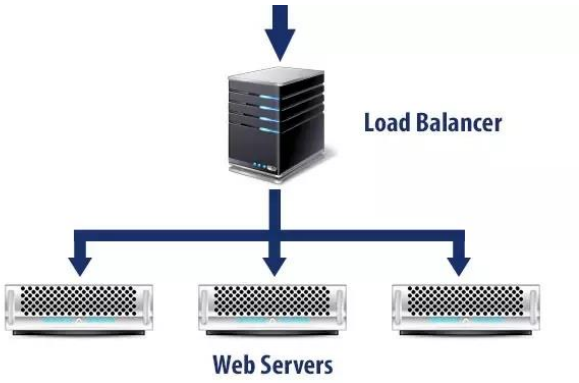
Create an AMI from the EC2 Instance

EC2 > Actions > Image > Create Image > [Done]

To view Created image : Go to Images > AMI

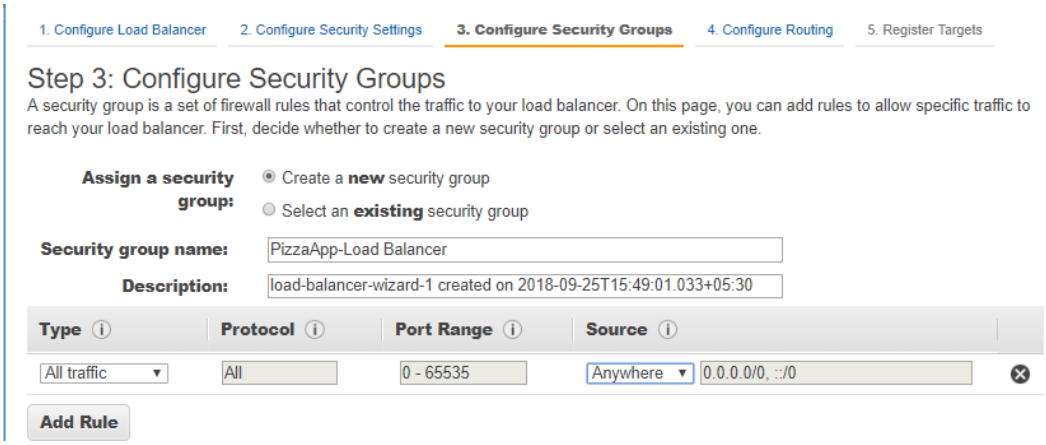
Create Load Balanacer

Load balancing is the process of distributing network traffic across multiple servers. This ensures no single server bears too much demand. By spreading the work evenly, load balancing improves application responsiveness



Creating Load Balancer for PizzaApp

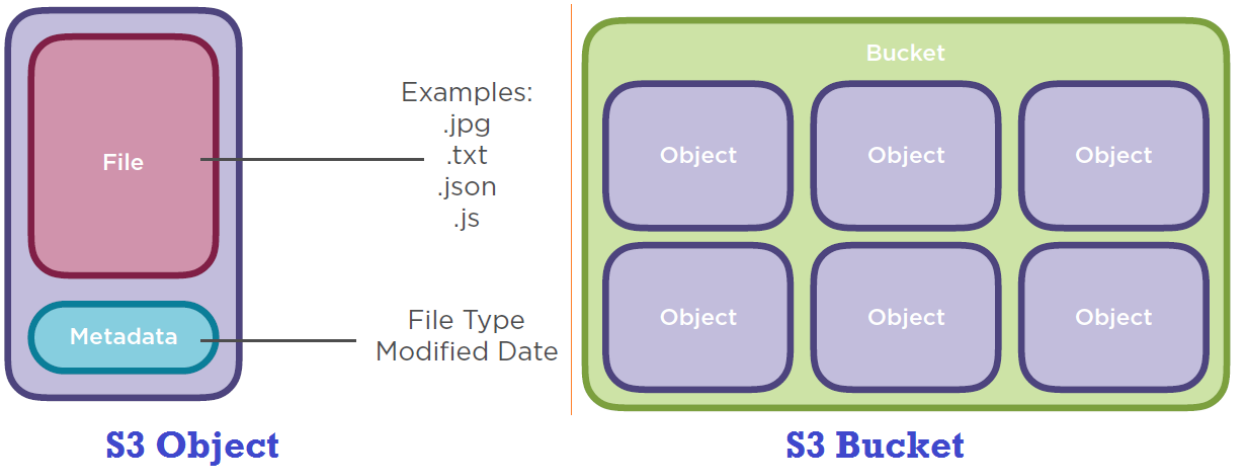
1.EC2 > left : LOAD BALANCING > Load Balancer > Create Load Balancer



5. S3 (Simple Storage Service)

S3 stores data in the form of Objects in the Region you specify

S3 Object = File + Metadata; **S3 Bucket** = Collection of Objects



S3 Bucket Example

- **Bucket Name**: pizza-luvrs
- **Region**: Oregon
- **URL**: s3-us-west-2.amazonaws.com/pizza-luvrs

S3 Object Key Example

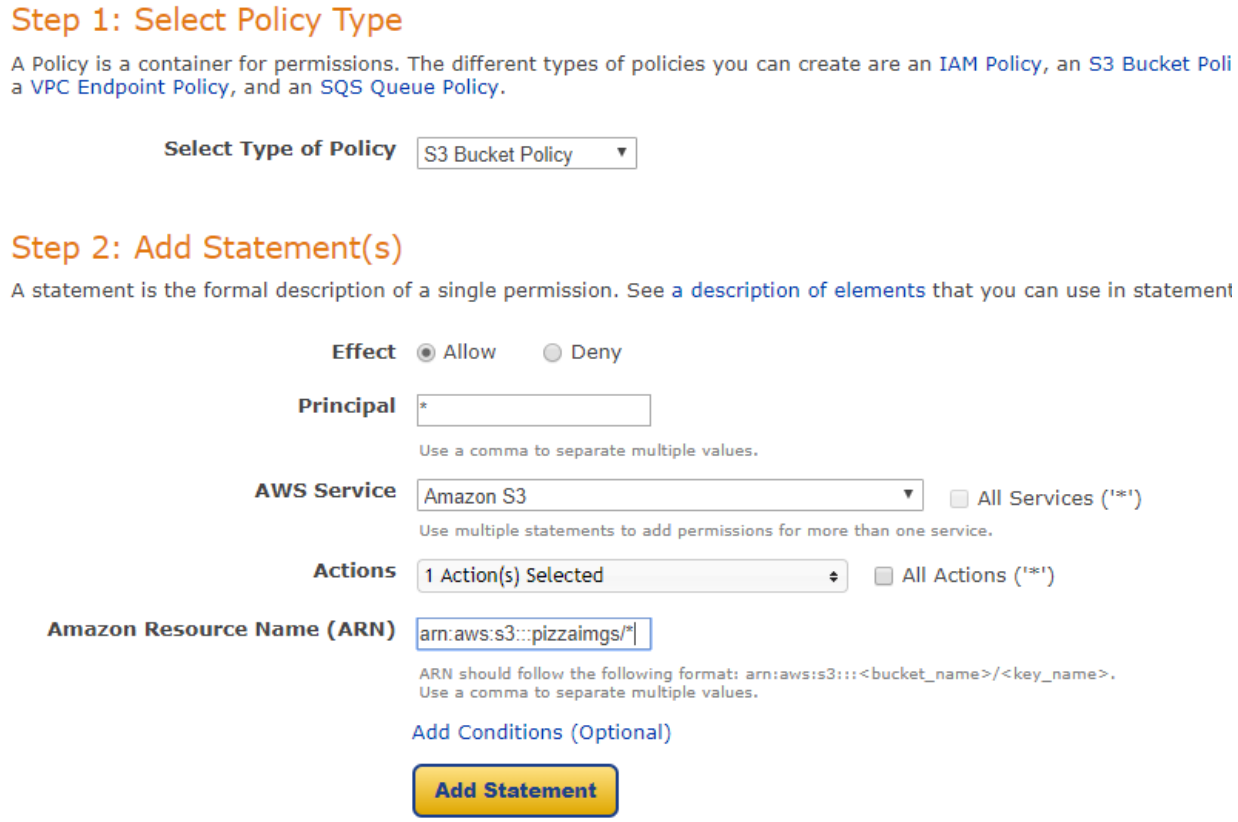
Path & filename is considered as KEY, key is unique

- **Filename**: dog.png
- **Folder Name**: images
- **Object Key**: images/dog.png

Task1: Create S3 Bucket for storing Pizza images

1.S3 > Create Bucket > Name : pizzaimgs

2.Add prtmissions to make this bucket as public. For this use
https://awspolicygen.s3.amazonaws.com/policygen.html & Copy JSON



3.Go to S3 Bucket > Permissions > Bucket Policy : Paste JSON > save

6. Databases (RDS & DynamoDB)

Relational Database Service

Managed database instances in AWS running on EC2

- Software Upgrades
- Nightly Database Backups
- Monitoring

RDS Backups

Multi-AZ Deployment Database replication to different Availability Zone

Automatic failover in case of catastrophic event

- Occurs daily
- Configurable backup window
- Backups stored 1 - 35 days
- Restore database from backup

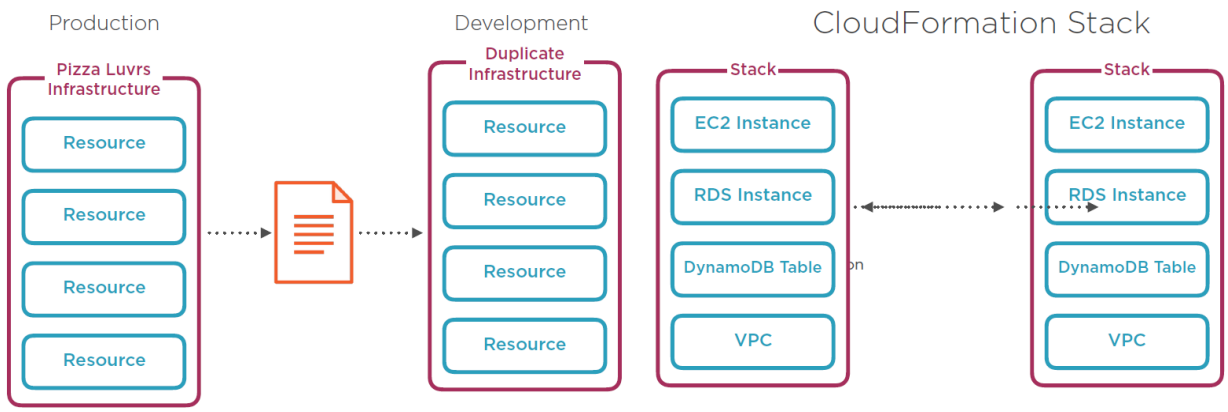
RDS Database Options

Task 1 : Creating a Database in RDS

1.Services > RDS > Create DataBase > Ex. PostgreSQL >

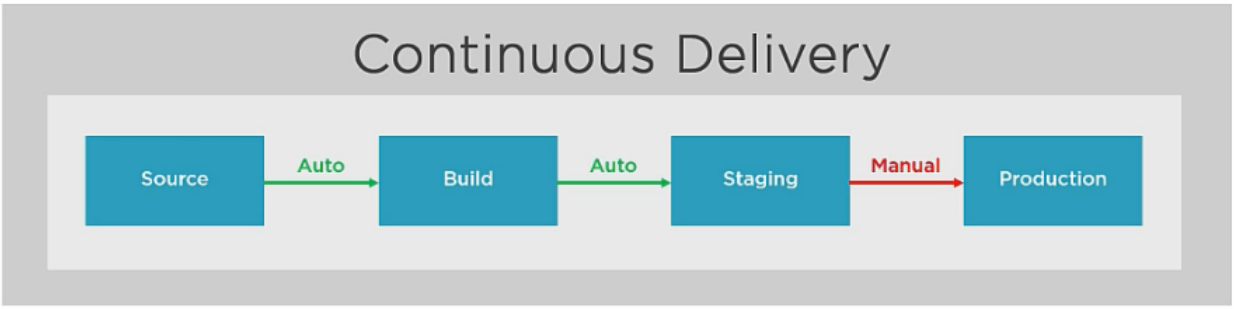
7. CloudFormation

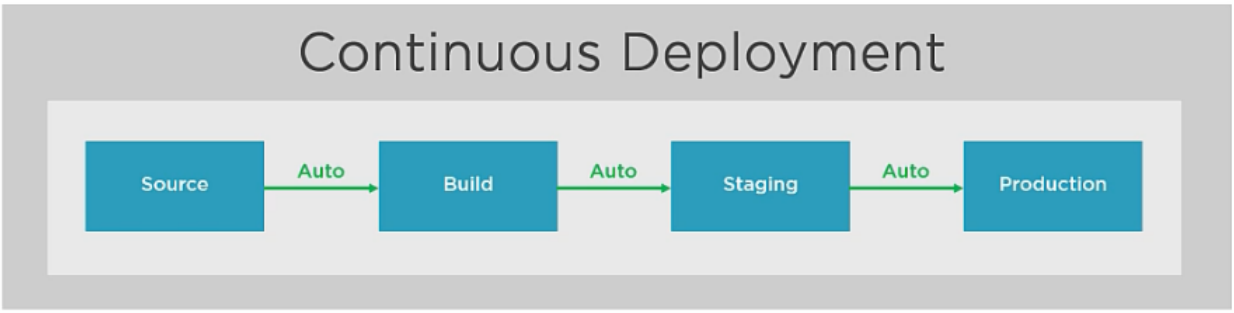
CloudFormation allows you to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all regions and accounts. This file serves as the single source of truth for your cloud environment.



AWS Certified DevOps Engineer

1.AWS Certified DevOps Engineer: Continuous Delivery and Automation





1.AWS CodeCommit

AWS CodeCommit is a secure, highly scalable, managed source control service that hosts private Git repositories. **AWS CodeCommit** eliminates the need for you to manage your own source control system or worry about scaling its infrastructure. You can use **AWS CodeCommit** to store anything from **code** to binaries.

```
>Get-AWSCredentials -ListProfiles
>Set-AWSCredentials -AccessKey AKIAJBJRAI3S3TZEETFA -SecretKey yGD05dNukCMt+oQw+so5
-StoreAs codecommit
>cd 'C:\Program Files (x86)\AWS Tools\CodeCommit\'
> .\git-credential-AWSS4.exe -p codecommit
```

IAM Sign in URL **: user/123abc****@**

To create Repository

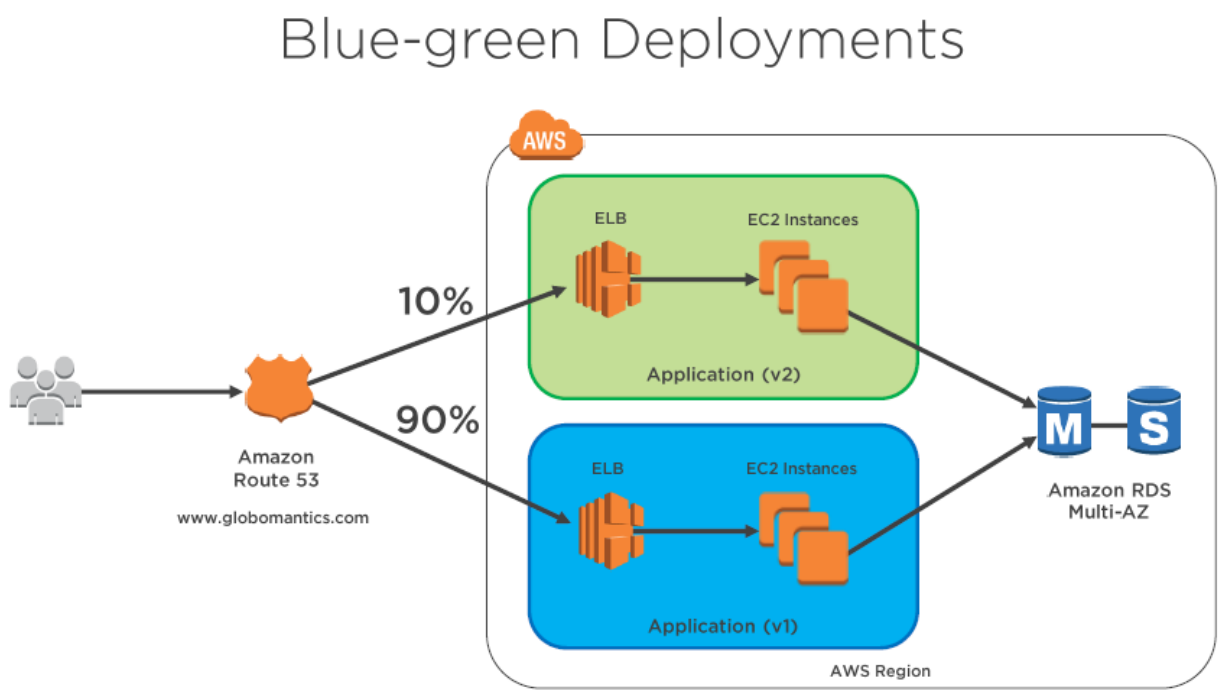
Services > CodeCommit > Create Repo > Name : "FirstRepo" > Create

Clone Repo:

```
git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/FirstRepo
```

2. AWS CodeDeploy

Code : https://github.com/mikepfeiffer/aws-codedeploy-linux

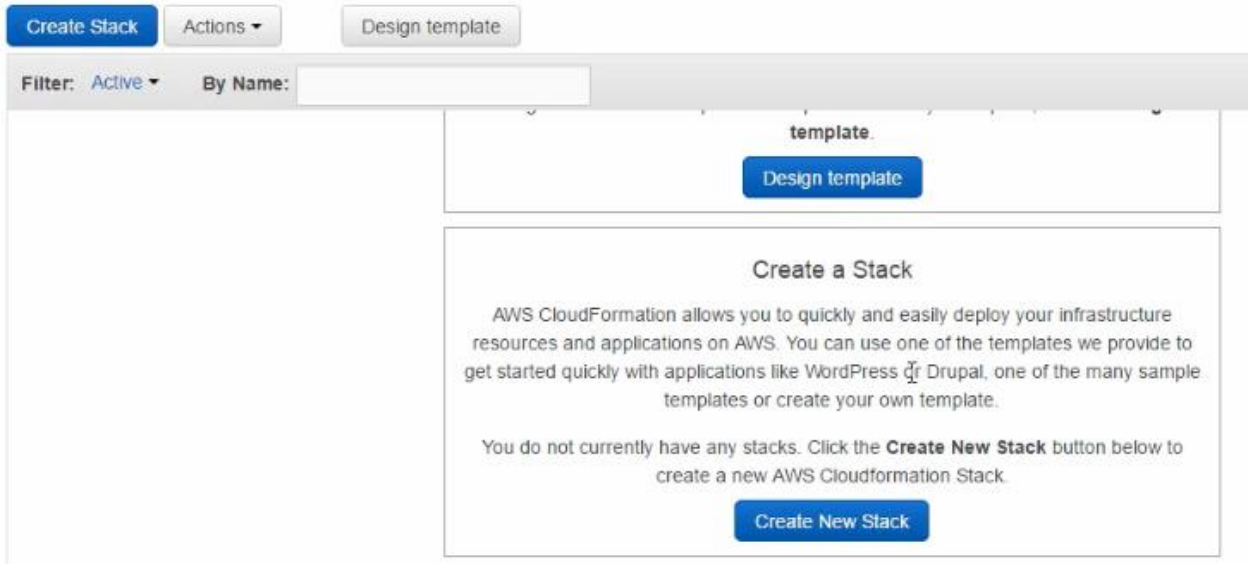


3.AWS CodePipeline

4.CloudFormation


```
{
    "AWSTemplateFormatVersion" : "2010-09-09",

    "Description" : "Single Instance",

    "Resources" : {
        "EC2Instance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "ImageId" : "ami-08111162",
                "InstanceType" : "t2.micro",
                "SecurityGroups" : "mysg",
                "KeyName"        : "virginia"
            }
        }
    }
}
```

```
"Resources" : {
    "Web1" : ... AWS::EC2::Instance ...,
    "WebSecurityGroup" : {
        "Type" : "AWS::EC2::SecurityGroup",
        "Properties" : {
            "GroupDescription" : "Enable HTTP and SSH access",
            "SecurityGroupIngress" : [
                {
                    "IpProtocol" : "tcp",
                    "FromPort"   : "80",
                    "ToPort"     : "80",
                    "CidrIp"     : "0.0.0.0/0"
                },
                {
                    "IpProtocol" : "tcp",
                    "FromPort"   : "22",
                    "ToPort"     : "22",
                    "CidrIp"     : "0.0.0.0/0"
                }
            ]
        }
    }
```



Select Template

Select the template that describes the stack that you want to create. A stack is a group of related resources that you manage as a single unit.

Design a template Use AWS CloudFormation Designer to create or modify an existing template. Learn more.

Design template

Choose a template A template is a JSON-formatted text file that describes your stack's resources and their properties. Learn more.

- Select a sample template
- Upload a template to Amazon S3

Choose File cloudformation.template

- Specify an Amazon S3 template URL

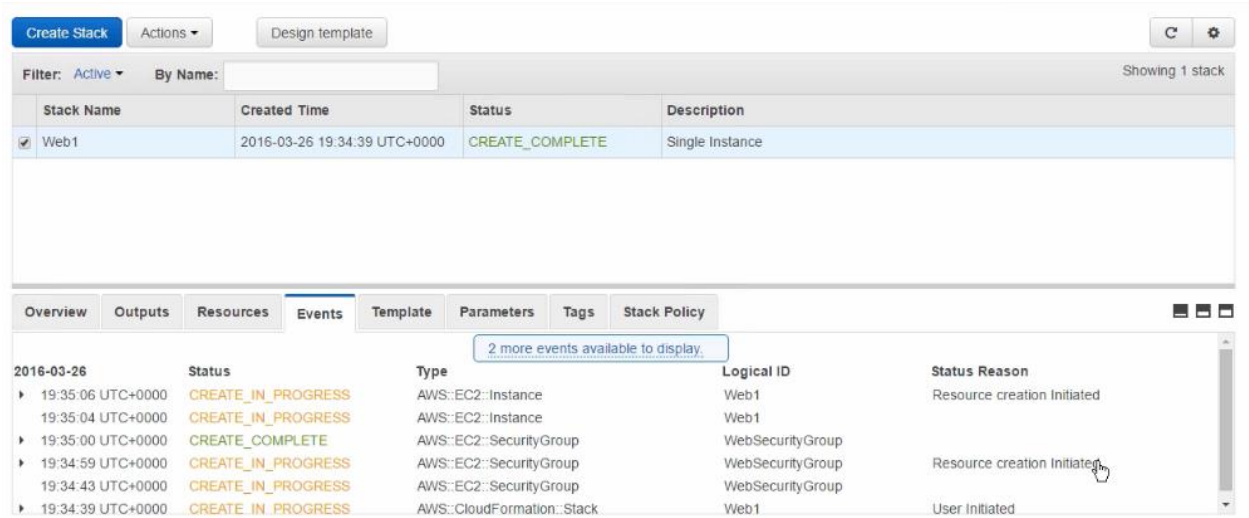
Cancel Next

Specify Details

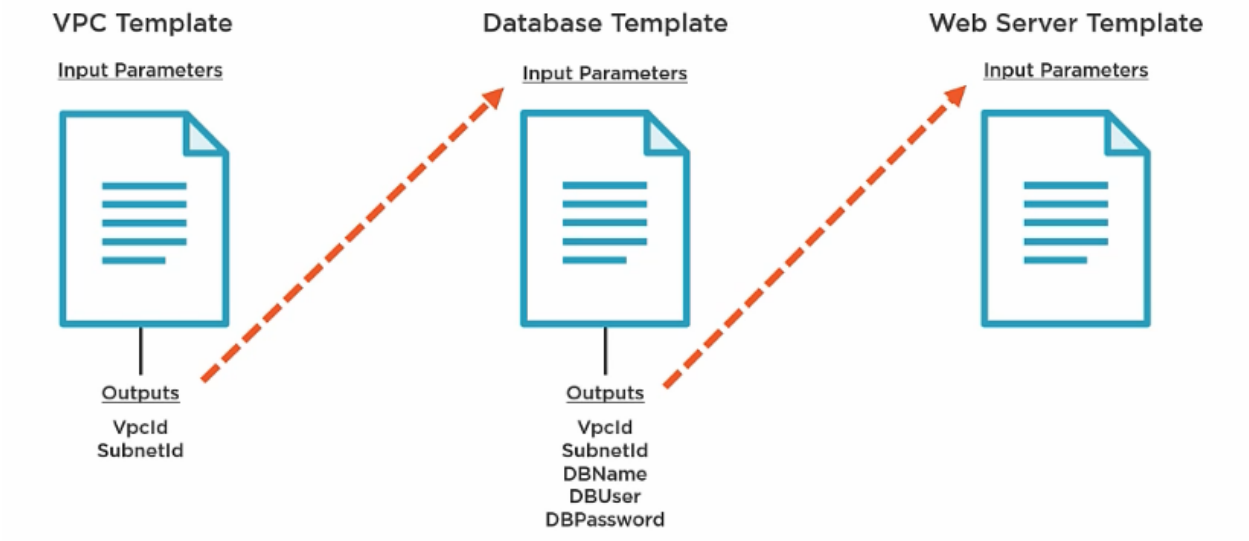
Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. Learn more.

Stack name Web1

Cancel Previous Next



Creating Nested Templates



5.Elastic BeanStalk

Platform-as-a-Service (PaaS)

Service for deploying and scaling web apps and services

Autoscaling, load balancing & health monitoring

Platform management and code deployment

All Applications

PhpApp

Phpapp-env

Environment tier: Web Server
Platform: PHP 7.2 running on 64bit Amazon Linux/2.8.1
Running versions: Sample Application
Last modified: 2018-09-25 20:03:56 UTC+0530
URL: Phpapp-env.pbnweqjmt3.us-east-1.elasticbeanstalk.c...

Working with Docker

Single Container Docker

Used to deploy a Docker image and source code to EC2 instances

Multicontainer Docker

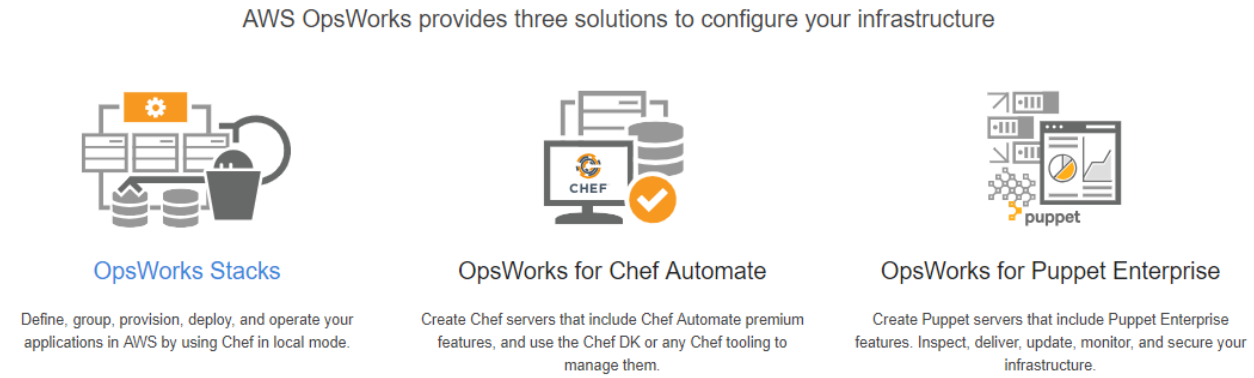
Used to deploy multiple Docker containers to an Amazon EC2 Container Service (ECS) cluster

Preconfigured Docker

Used with popular software stacks such as Java with GlassFish, Go, Python

6.OpsWorks

AWS OpsWorks is a configuration management service that helps you build and operate highly dynamic applications, and propagate changes instantly.

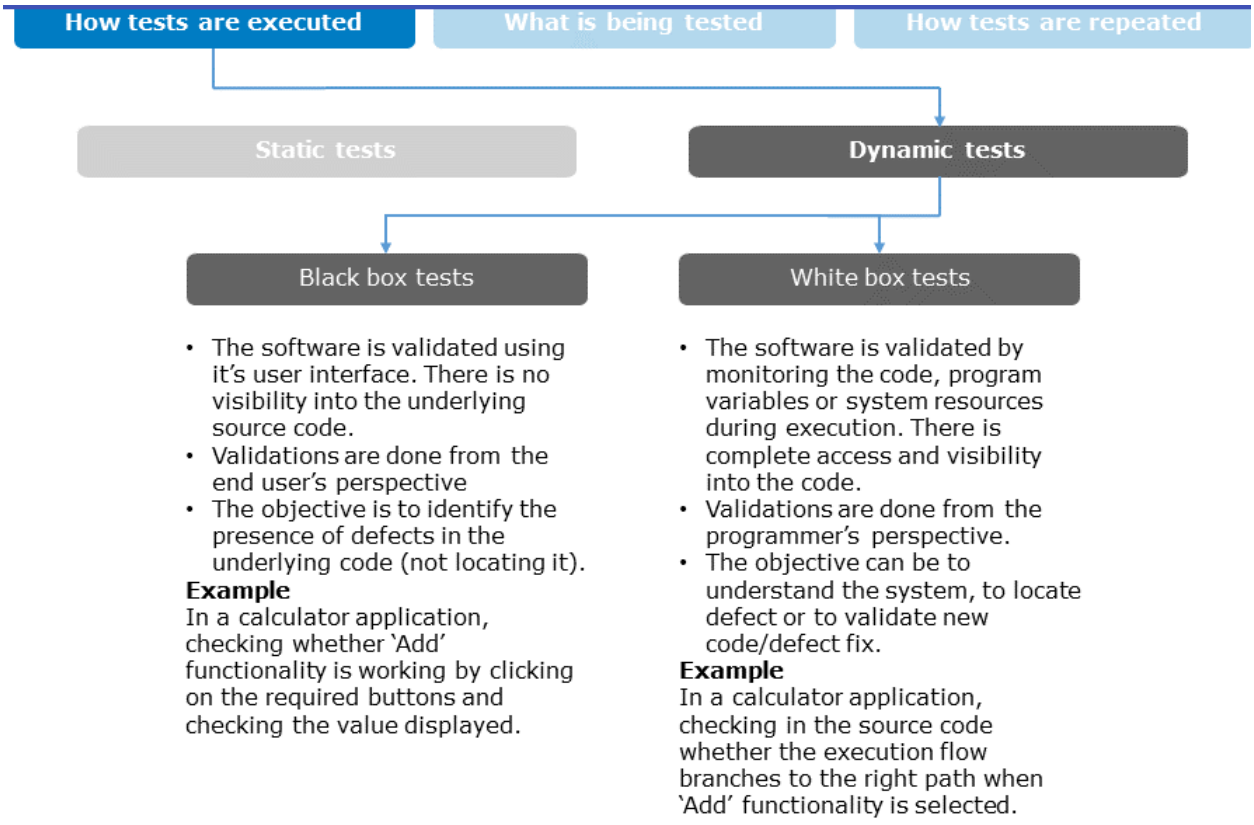


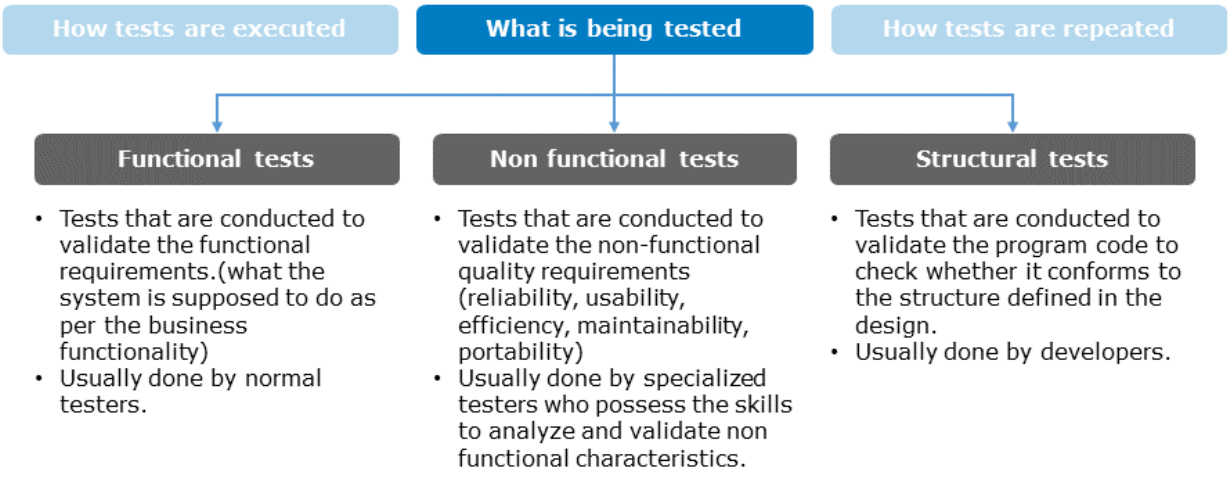
7. CloudWatch

Software testing is a set of processes and tasks that take place throughout the software development life cycle. It helps to reduce the risk of failures that may occur during operational use and, thus, ensure the quality of the software system.

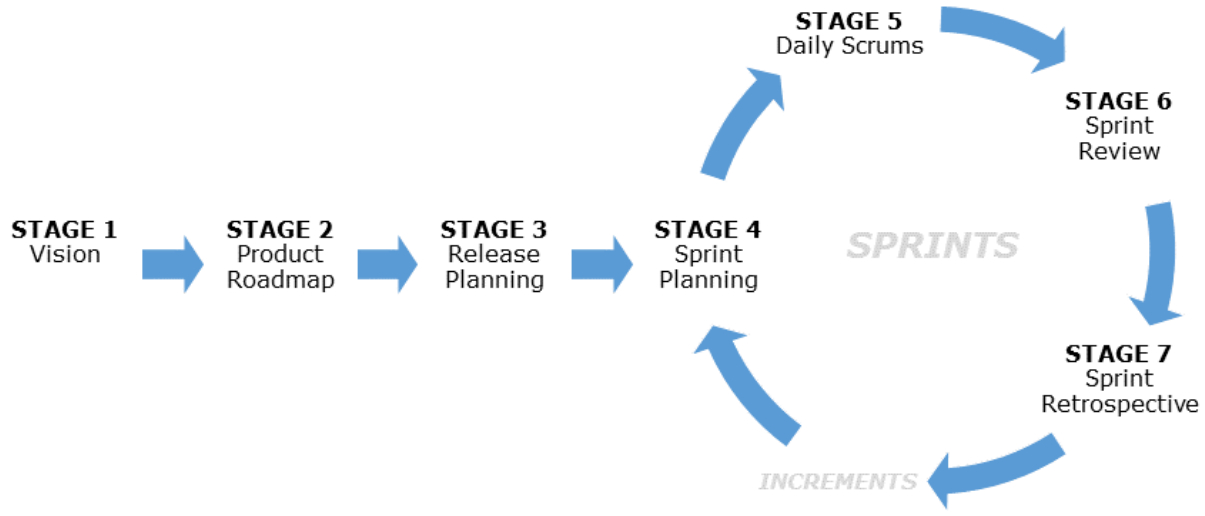
Objectives of software testing

- Preventing defects from entering the system.
- Finding defects existing in the system.
- Measuring the quality of the system.

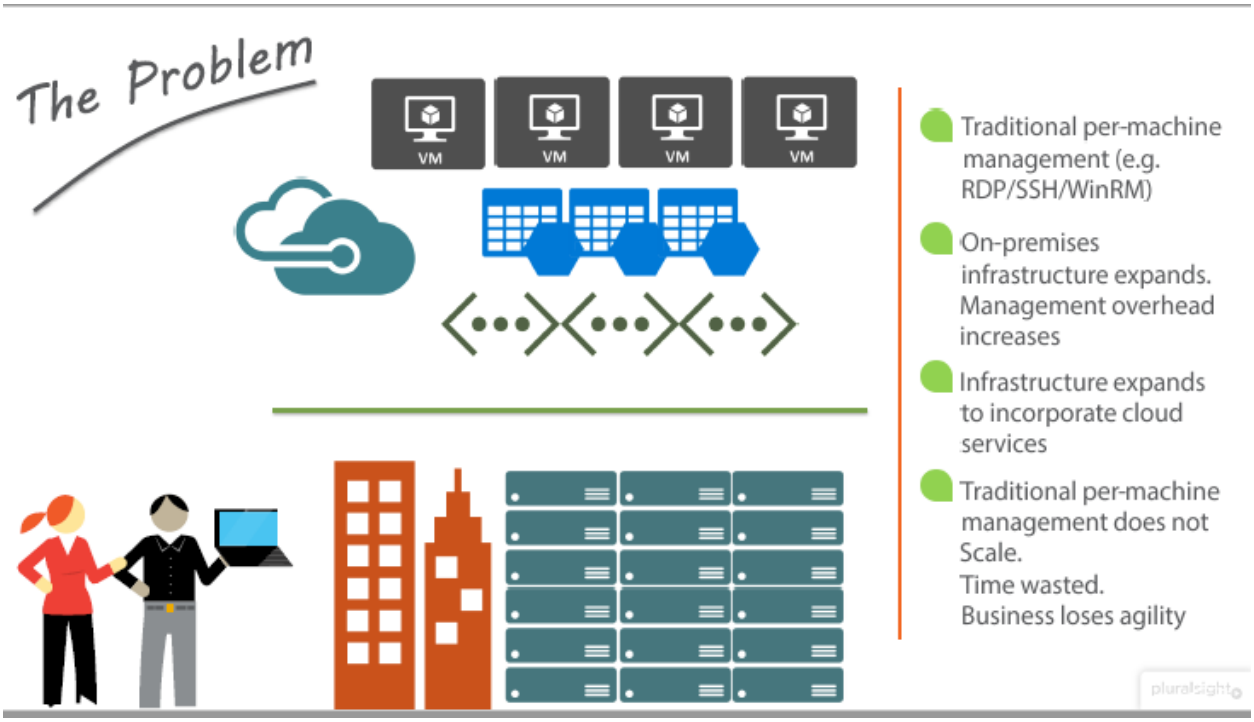




Level	Objective	Test Basis	Done By
Component	Used to test the functional and non-functional characteristic of the code/module.	• Component Requirement • Code	White Box Testers
Component-Integration	Used to test the interaction between software components and is done after component testing.	• Software and System design. • Architecture • Use Cases	White Box Testers
System	Used to test the functional and non-functional characteristics of the complete system.	• System and Software requirement Specification. • Use Cases	Black Box Testers
System-Integration	Used to test the interaction between different systems or between hardware and software.	• System and Software requirement Specification. • Use Cases • Software and System design.	Black Box Testers
UAT – Alpha	Also called as factory acceptance testing. It is done at developing organizations site but not by the development team.	• System Requirements • User Requirements • Use Cases	Users / Stakeholders
UAT - Beta	Also called as field/site acceptance testing is performed by customers at their own locations.	• System Requirements • User Requirements • Use Cases	Users / Stakeholders



8. Chef - Automate IT Infrastructure



Infrastructure as Code

Versionable | Testable | Repeatable

Hosted Chef

- Fully cloud hosted (SaaS)
- Easy to set up
- No ongoing maintenance
- Very limited customization options
- No control over access speeds

Chef Server

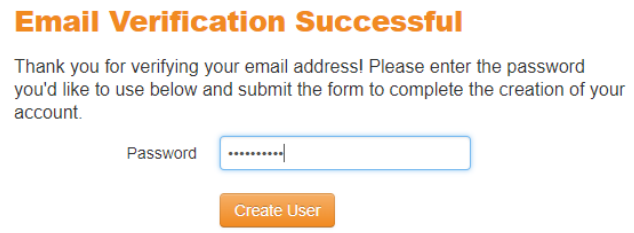
- Installed on-premises or cloud IaaS
- More complex installation
- Requires normal maintenance
- Allows fine-grained control
- Enables rapid deployment/updates

Idempotent : unchanged in result by multiple, many actions

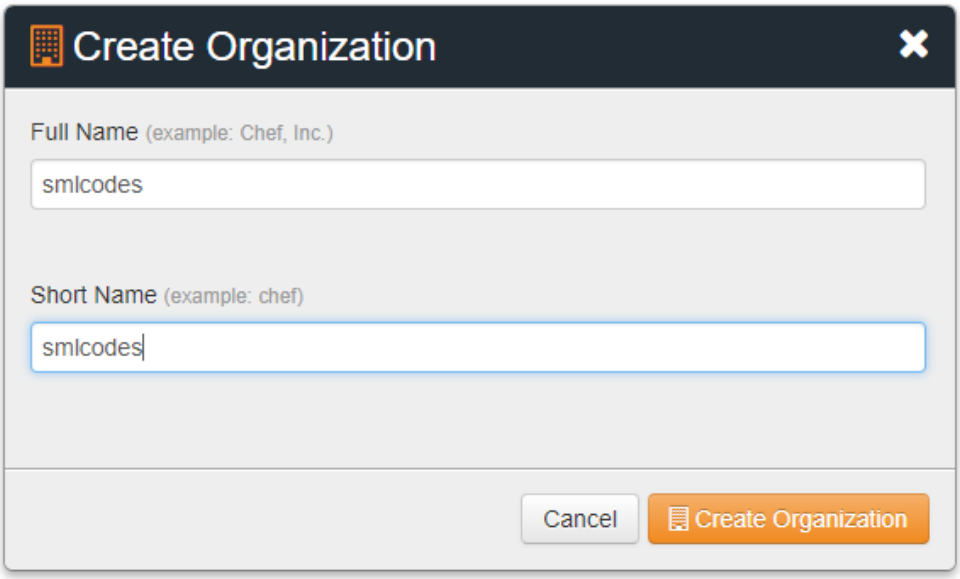
Signup for HostedChef

1.Chef.io / https://getchef.opscode.com/signup > Get Chef

2.After Vefify email, set password



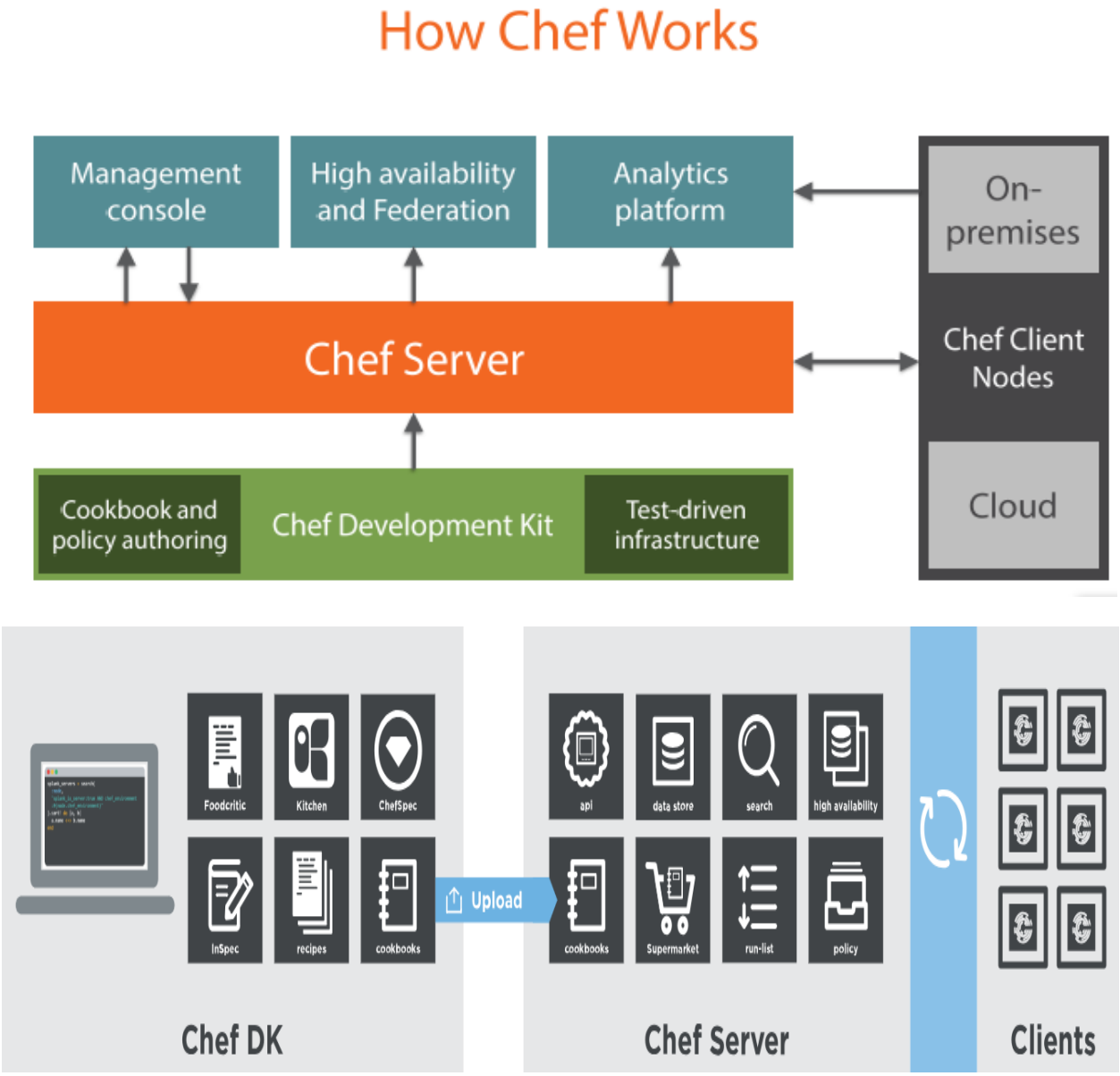
3.Create new Organization



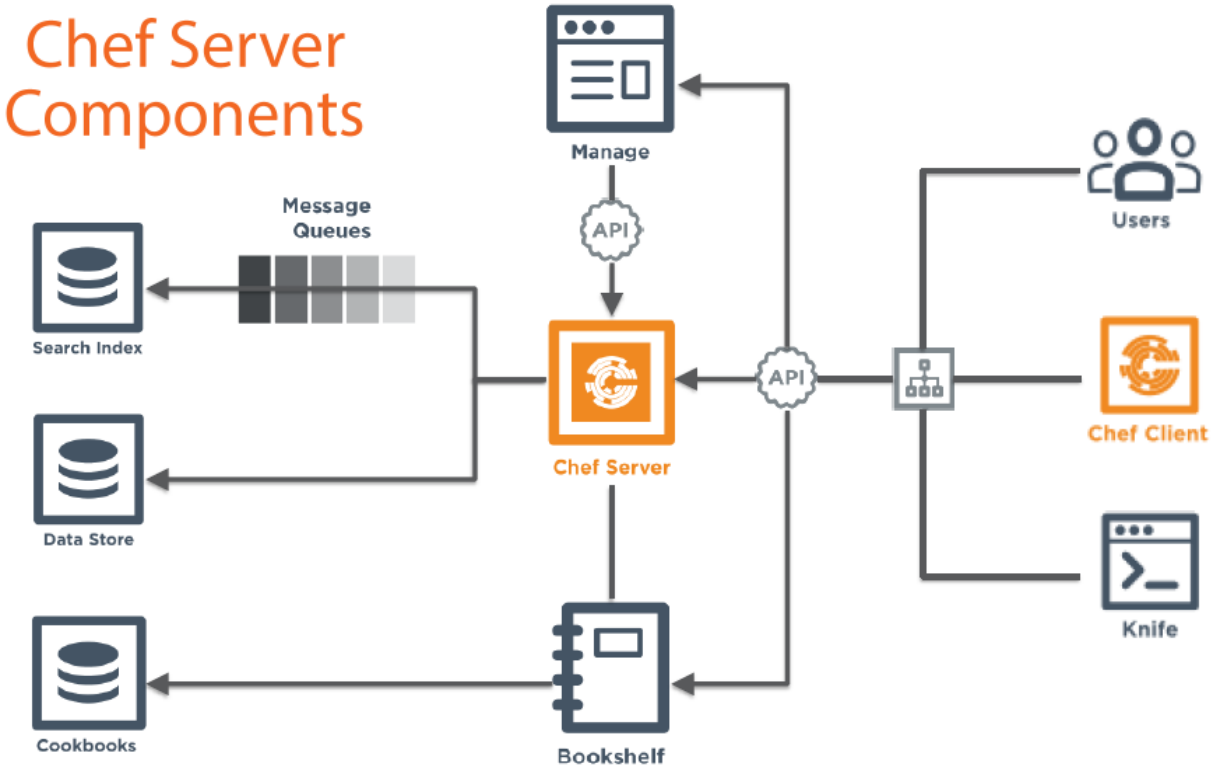
Chef is a configuration management tool. Trying to coordinate the work of multiple system administrators and developers involving hundreds, or even thousands, of servers and applications to support a large customer base is complex and typically requires the support of a tool.

Examples of modern IT configuration management tools are CFEngine, Puppet, the Desired State Configuration engine in Microsoft Windows, Ansible, SaltStack, and of course, Chef.

Architecture



- The **Chef DK workstation** is the location where users interact with Chef. On the workstation users author and test cookbooks using tools such as Test Kitchen and interact with the Chef server using the knife and chefcommand line tools.
- The **Chef server** acts as a hub for configuration data. The Chef server stores cookbooks, the policies that are applied to nodes, and metadata that describes each registered node that is being managed by Chef.
- **Chef client nodes** are the machines that are managed by Chef. The Chef client is installed on each node and is used to configure the node to its desired state.



Once you master Chef, you can use it to

- Fully automate deployments, including internal development and end-user systems
- Automate scaling of infrastructure
- Make your infrastructure self-healing

Installing

1. download an Ubuntu 14.04 box using vagrant. To create an environment, inside your folder run init command, it will create **'vagrantfile'**

```
vagrant init ubuntu/trusty64
```

2.Navigate to vagrantfile creaged folder & Start the Environmet

```
$ vagrant up
```

3. connect to Environment

```
$ vagrant ssh
```

4. Install the Chef DK on your Ubuntu instance

The Chef DK provides tools that enable you to manage your servers remotely from your workstation. But it also provides tools that allow you to configure a machine directly.

To use proxy : `sudo http_proxy=http://10.219.2.220:80 apt-get update`

Check CURL installation

```
sudo apt-get updatesudo apt-get -y install curl
```

To Set Shared folder, edit vagrantfile as

```
config.vm.synced_folder "D:\\DevOps\\Instl\\VagrantBoxes\\SyncFolder", "/vagrant"
```

we named our syncd folder as "vagrant", you can find the files in Syncfolder by going /vagrant/

```
vagrant@vagrant-ubuntu-trusty-64:~$ cd /
vagrant@vagrant-ubuntu-trusty-64:/$ ls
bin   dev  home        lib    lost+found  mnt  proc  run   srv  tmp  vagrant  vmlinuz
boot  etc  initrd.img  lib64  media       opt  root  sbin  sys  usr  var
vagrant@vagrant-ubuntu-trusty-64:/$ cd vagrant/
vagrant@vagrant-ubuntu-trusty-64:/vagrant$ ls
chefdk_3.2.30-1_amd64.deb
```

3. Install ChefDK:

```
sudo http_proxy=http://10.219.2.220:80 apt-get update
sudo dpkg --configure -a
sudo dpkg -i chefdk_*.deb
```

Handson

1.Downloaded chefdk in Normal ubuntu System & installed

2.Check chef -v version details

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef$ chef -v
Chef Development Kit Version: 3.2.30
chef-client version: 14.4.56
delivery version: master (6862f27aba89109a9630f0b6c6798efec56b4efe)
berks version: 7.0.6
kitchen version: 1.23.2
inspec version: 2.2.70
```

1.Creating Recipe in Chef

Chef writen in Ruby. Now we are going to create a recipe **"hello.rb"** , here .rb is Ruby extension

Collection of Recipes are called CookBoook

1.Create chef_repo folder, go into it

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$
```

2.Create a resource in System using Chef

create hello.rb file inside **chef_repo**

```
file 'motd' do
       content 'Hello, World'
end
```

3.Apply recipe to Current System.

The above Code means, create a resorce(fille) with name “motd” with content as “hello world”

By executing **chef-apply hello.rd,** Chef will create a new file in our current infrasrure system

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ chef-apply hello.rb
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * file[motd] action create
    - create new file motd
    - update content in file motd from none to 03675a
    --- motd    2018-09-26 22:42:58.885793277 +0530
    +++ ./.chef-motd20180926-23053-ysyfqd       2018-09-26 22:42:58.885793277 +0530
    @@ -1 +1,2 @@
    +Hello, World
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ ls
hello.rb  motd
```

4.Here chef is idempotent, that means multiple changes won’t change the result.

It we change the hello.rd file, chef wont consider the changes until we apply changes

5.Delete file, using action

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ gedit hello.rb

file 'motd' do
       action:delete
end

satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ chef-apply hello.rb
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * file[motd] action delete
    - delete file motd
```

Complete Log

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ gedit hello.rb
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ cat hello.rb
file &apos;motd&apos; do
       content &apos;Hello, World&apos;
end
```

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ ls
hello.rb
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ chef-apply hello.rb
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * file[motd] action create
    - create new file motd
    - update content in file motd from none to 03675a
    --- motd    2018-09-26 22:42:58.885793277 +0530
    +++ ./.chef-motd20180926-23053-ysyfqd       2018-09-26 22:42:58.885793277 +0530
    @@ -1 +1,2 @@
    +Hello, World
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ ls
hello.rb  motd
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ gedit hello.rb
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ chef-apply hello.rb
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * file[motd] action delete
    - delete file motd
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$
```

2. Chef : Recipes

1.Create a Resouce : Install a Software in Host Machine

In above ex, we just created a file as a resorce in host mechine. Now go to more advance install Softtware as a pkg in host machine

- Apache2 pkg should install in host mechine
- Apache2 Should enable & Auto Start
- Create index.html, & make it as apache Homepage

1.edit hello.rd to perform above 3 steps on host system

```
package 'apache2'

service 'apache2' do
       action[:enable, :start]
end

file '/var/www/html/index.html' do
       content '<h1>Hello, Chef!!</h1>'
end
```

2.Do chef-apply

```
Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ sudo chef-apply hello.rb
[sudo] password for satya:
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[apache2] action install

    - install version 2.4.29-1ubuntu4.1 of package apache2
```

```
  * service[apache2] action enable (up to date)
  * service[apache2] action start (up to date)
  * file[/var/www/html/index.html] action create
    - update content in file /var/www/html/index.html from b66332 to c0086c
    --- /var/www/html/index.html        2018-09-26 23:24:20.577635660 +0530
    +++ /var/www/html/.chef-index20180926-27935-qxmg32.html     2018-09-26
23:24:28.713710303 +0530
    @@ -1,376 +1,2 @@
```

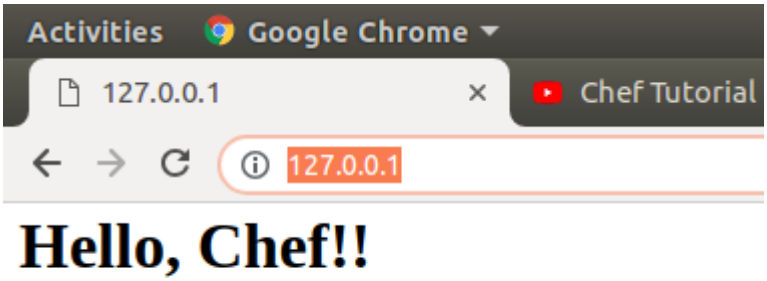
Cross Checking

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ sudo chef-apply hello.rb
Recipe: (chef-apply cookbook)::(chef-apply recipe)
  * apt_package[apache2] action install (up to date)
  * service[apache2] action enable (up to date)
  * service[apache2] action start (up to date)
  * file[/var/www/html/index.html] action create (up to date)
```

3.Staring apache Server

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ service apache2
startsatya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$
```

4.Test by going : http://127.0.0.1/



Or using Curl

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef_repo$ curl localhost
<h1>Hello, Chef!!</h1>
```

---

3. Chef : Cookbooks

We are creating cookbooks folder, for working

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/cookbooks$ PS1='\W>';
```

shorten file path in terminal PS1='\W> '

To generate CookBook Syntax is

```
chef generate cookbook <Cookbook_Name>
```

1.Createing Apache Cookbook

```
cookbooks> chef generate cookbook chef_apache2
```

Here a folder is created inside /cookbooks withthe name chef_apache2

```
cookbooks> ls
chef_apache2
cookbooks> cd chef_apache2/
chef_apache2> tree
.
├── Berksfile
├── CHANGELOG.md
├── chefignore
├── LICENSE
├── metadata.rb
├── README.md
├── recipes
│   └── default.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
│           └── default_spec.rb
└── test
    └── integration
        └── default
            └── default_test.rb

7 directories, 10 files
```

2.Create a Template called index.html in chef_apache2 cookbook

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/cookbooks$
chef generate template chef_apache2 index.html

Recipe: code_generator::template
  * directory[./chef_apache2/templates] action create
    - create new directory ./chef_apache2/templates
  * template[./chef_apache2/templates/index.html.erb] action create
    - create new file ./chef_apache2/templates/index.html.erb
    - update content in file ./chef_apache2/templates/index.html.erb from none to
e3b0c4
```

here new folder (template) inside **chef_apache2** and places index.html inside template folder

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/cookbooks/chef_apache2$ tree
.
├── Berksfile
├── CHANGELOG.md
├── chefignore
├── LICENSE
├── metadata.rb
```

```
├── README.md
├── recipes
│   └── default.rb
├── spec
│   ├── spec_helper.rb
│   └── unit
│       └── recipes
│           └── default_spec.rb
├── templates
│   └── index.html.erb
└── test
    └── integration
        └── default
            └── default_test.rb
```

3.Write “hello world” inside index.html

```
chef_apache2$ gedit templates/index.html.erb
```

4.edit recipe file recipes/default.rb to perform below 3 steps on host system

- Apache2 pkg should install in host mechine
- Apache2 Should enable & Auto Start
- Create index.html, & make it as apache Homepage

```
chef_apache2$ gedit recipes/default.rb

package 'apache2'

service 'apache2' do
       action [:enable, :start]
end

template '/var/www/html/index.html' do
       source 'index.html.erb'
end
```

5.Applay Cookbook to local System

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef$
 sudo chef-client --local-mode --runlist ‘recipe[chef_apache2]’;
[sudo] password for satya:
[2018-09-27T20:53:43+05:30] WARN: No config file found or specified on command line,
using command line options.
Starting Chef Client, version 14.4.56
resolving cookbooks for run list: ["chef_apache2"]
Synchronizing Cookbooks:
  - chef_apache2 (0.1.0)
Installing Cookbook Gems:
Compiling Cookbooks...
Converging 3 resources
```

```
Recipe: chef_apache2::default
  * apt_package[apache2] action install (up to date)
  * service[apache2] action enable (up to date)
  * service[apache2] action start (up to date)
  * template[/var/www/html/index.html] action create
    - update content in file /var/www/html/index.html from c0086c to 95dfba
    --- /var/www/html/index.html        2018-09-26 23:24:28.713710303 +0530
    +++ /var/www/html/.chef-index20180927-5228-95sw5q.html      2018-09-27
20:53:46.556287854 +0530
    @@ -1,2 +1,2 @@
    -<h1>Hello, Chef!!</h1>
    +Hello Satya

Running handlers:
Running handlers complete
Chef Client finished, 1/4 resources updated in 03 seconds
```

Check Apache Homepage

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef$ curl localhost
Hello Satya
```

4. CookBooks + Chef Server + Nodes

So far we did changes on local system only.

Now we are going to execute our cookbooks in Hosted Nodes through Chef server

```
                                  -----------------
                                  |  Chef         |        ---------- Node 1
[CookBooks]                ---->  |  Server       |        ---------- Node 2
(Programmer Mechine)              |               |        ---------  Node 3
                                  -----------------
```

Chef Server : Create Online Chef server

1.Create Chef server by going Hosted Chef Server - Manage Chef

2.Login to Chef Server : https://manage.chef.io/

3.Administration > Oraganization >smlcodes > Actions > Starter Kit > Download Starter Kit

4.Extract Downloaded zip (**Desktop/DevOps/chef/chef-starter**)

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef-starter$ tree
.
└── chef-repo
    ├── cookbooks
    │   ├── chefignore
    │   └── starter
    │       ├── attributes
    │       │   └── default.rb
    │       ├── files
    │       │   └── default
    │       │       └── sample.txt
    │       ├── metadata.rb
    │       ├── recipes
    │       │   └── default.rb
    │       └── templates
    │           └── default
    │               └── sample.erb
    ├── README.md
    └── roles
        └── starter.rb

10 directories, 8 files
```

5.https://supermarket.chef.io/ where all the user created cookbooks are stored.

For example Search for learn_chef_apache2,

we are going to use above cookbook in this example

6. Download cookbook using Knife

i. Go to ***/DevOps/chef/chef-starter/chef-repo*** & run **knife** command to download cookbook.

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef-starter/chef-repo$

knife cookbook site download learn_chef_apache2

Downloading learn_chef_apache2 from Supermarket at version 0.3.0 to
/home/satya/Desktop/DevOps/chef/chef-starter/chef-repo/learn_chef_apache2-0.3.0.tar.gz
Cookbook saved: /home/satya/Desktop/DevOps/chef/chef-starter/chef-
repo/learn_chef_apache2-0.3.0.tar.gz
```

ii.Exatract tar file to cookbooks folder

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef-starter/chef-repo$ ls
cookbooks  learn_chef_apache2-0.3.0.tar.gz  README.md  roles
```

```
satya@satya-Aspire-E5-523:~/Desktop/DevOps/chef/chef-starter/chef-repo$

tar -zxxvf learn_chef_apache2-0.3.0.tar.gz -C cookbooks

learn_chef_apache2/
learn_chef_apache2/.kitchen.yml
learn_chef_apache2/Berksfile
learn_chef_apache2/Berksfile.lock
learn_chef_apache2/chefignore
learn_chef_apache2/metadata.json
learn_chef_apache2/metadata.rb
learn_chef_apache2/README.md
learn_chef_apache2/recipes/
learn_chef_apache2/templates/
learn_chef_apache2/templates/default/
learn_chef_apache2/templates/default/index.html.erb
learn_chef_apache2/recipes/default.rb
```

After Extraction remove .tar file

```
rm -rf learn_chef_apache2-0.3.0.tar.gz
```

Check ***learn_chef_apache2/recipes/default.rb*** it contains same code we configue before

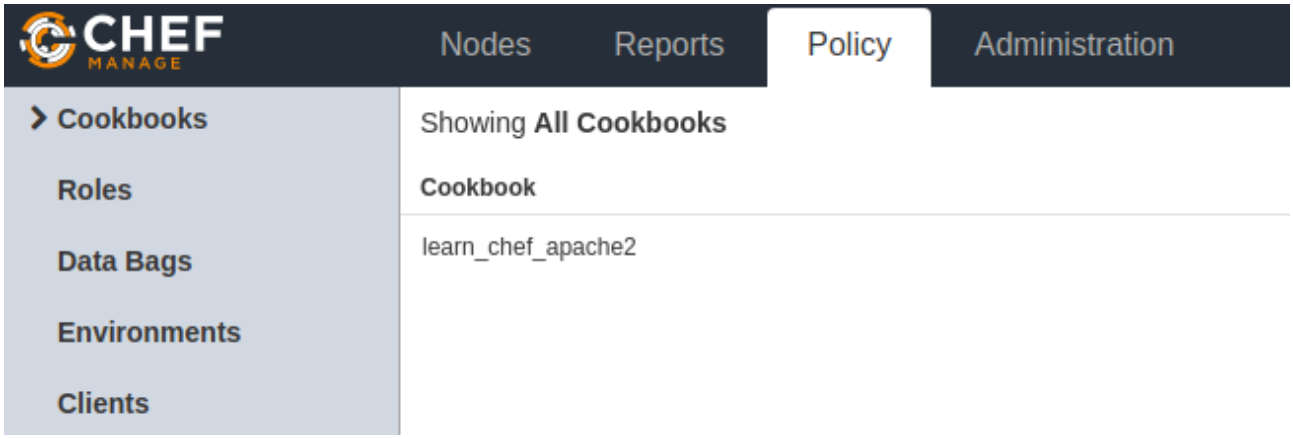
```
apt_update 'Update the apt cache daily' do
  frequency 86_400
  action :periodic
end

package 'apache2'

service 'apache2' do
  supports :status => true
  action [:enable, :start]
end

template '/var/www/html/index.html' do
  source 'index.html.erb'
end
```

7.Upload CookBook to Chef Server

go to Desktop/DevOps/chef/chef-starter/chef-repo folder & upload cookbook to chef server

```
chef-repo> knife cookbook upload learn_chef_apache2
Uploading learn_chef_apache2 [0.3.0]
Uploaded 1 cookbook.
```

8.Check Uploaded CookBook in server > policy tab



Now we are going to manage our nodes throgh Chef Server

Nodes : Manage Nodes using Chef server

We have two nodes created using vagrant, up them[**vagrant up, vagrant ssh**]

1. Ubuntu : **vagrant@vagrant-ubuntu-trusty-64:**
2. CentOS : **[vagrant@localhost ~]$**

IpAddress : check ip's by ping <ip>

change the password using `sudo passwd ubuntu` (by default `ubuntu` user has sudo-permissions with `NOPASSWD` set)

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"
  config.vm.network "forwarded_port", guest: 80, host: 5555
  config.vm.network "public_network"
end
```

1) wlp3s0 --> choose this
2) enp2s0f1

Ubuntu : 5555

- IpAddrees - **192.168.0.105**
- Username/pwd - ubuntu/ ubuntu

CentOs:6666

- IpAddrees - inet **192.168.0.107**
- Username/pwd - root / vagrant

1.Working with Ubuntu Node (192.168.0.105)

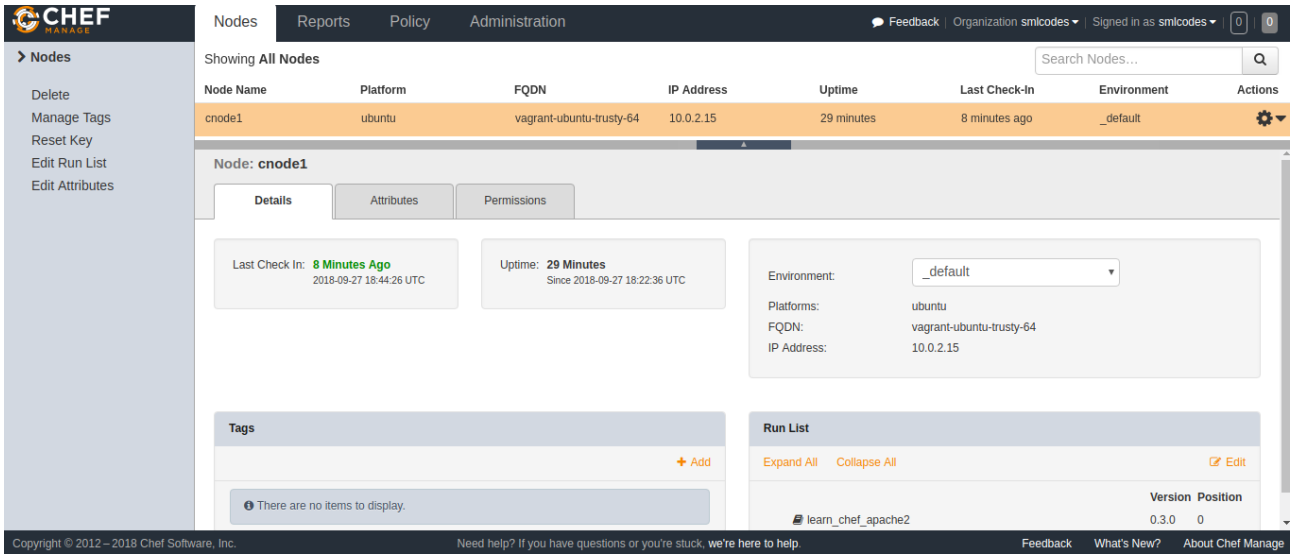
1.Go to Clinet workstation coomandline

```
/Desktop/DevOps/chef/chef-starter/chef-repo
```

2.Boostrap Node1 – Ubuntu & run cookbook on Node1

```
chef-repo> knife bootstrap 192.168.0.105 --ssh-user ubuntu --ssh-password ubuntu --sudo
--use-sudo-password --node-name cnode1 --run-list
&amp;apos;recipe[learn_chef_apache2]&amp;apos;
Node cnode1 exists, overwrite it? (Y/N) y
Client cnode1 exists, overwrite it? (Y/N) y
Creating new client for cnode1
Creating new node for cnode1
Connecting to 192.168.0.105
192.168.0.105 -----> Existing Chef installation detected
192.168.0.105 Starting the first Chef Client run...
192.168.0.105 Starting Chef Client, version 11.8.2
192.168.0.105 resolving cookbooks for run list: ["learn_chef_apache2"]
192.168.0.105 Synchronizing Cookbooks:
192.168.0.105   - learn_chef_apache2
192.168.0.105 Compiling Cookbooks...
192.168.0.105 Converging 3 resources
192.168.0.105 Recipe: learn_chef_apache2::default
192.168.0.105   * package[apache2] action install
192.168.0.105     - install version 2.4.7-1ubuntu4.20 of package apache2
192.168.0.105
192.168.0.105   * service[apache2] action enable
192.168.0.105     - enable service service[apache2]
192.168.0.105
192.168.0.105   * service[apache2] action start (up to date)
192.168.0.105   * template[/var/www/html/index.html] action create
192.168.0.105     - update content in file /var/www/html/index.html from 538f31 to
ef4ffd
192.168.0.105         --- /var/www/html/index.html      2018-09-27 18:46:21.787423744
+0000
192.168.0.105         +++ /tmp/chef-rendered-template20180927-2085-27sx61       2018-
09-27 18:46:35.617936153 +0000
192.168.0.105         @@ -1,379 +1,6 @@
```

3.Now we can check, knife automatically register Ubuntu Node1 , with Chef Server



4.Go To Node1-Ubuntu Terminal, Check the Home page

```
vagrant@vagrant-ubuntu-trusty-64:~$ curl localhost
<html>
  <body>
    <h1>hello world</h1>
  </body>
</html>
```

2.Working with Node 2 - CentOS (192.168.0.107)

1.Go to Clinet workstation coomandline

```
/Desktop/DevOps/chef/chef-starter/chef-repo
```

2.Boostrap Node2 – CentOs & run cookbook on Node2

```
chef-repo> knife bootstrap 192.168.0.107 --ssh-user vagrant --ssh-password vagrant --
sudo --use-sudo-password --node-name cnode2 --run-list
&apos;recipe[learn_chef_apache2]&apos;
```

`rescue in new_session': Authentication failed for user vagrant@192.168.0.107@192.168.0.107 (Net::SSH::AuthenticationFailed)
ERROR: Net::SSH::AuthenticationFailed: Authentication failed for user vagrant@192.168.0.107@192.168.0.107

Fixed it!

So when you are using hosted chef you need to pass in a private key with the bootstrap and have the public key in your autherized_keys file....

1. install the ChefSDK
2. SCP your starter kit from hosted Chef
3. extract the starter kit to `~/chef-repo`
4. generate a new keypair: `ssh-keygen`
5. add the public key to your autherized_keys file: `$ cat id_rsa.pub >> authorized_keys`
6. run the knife bootstrap with the following:

`sudo knife bootstrap {{server-ip}} --ssh-user {{your-server-user}} -i ~/.ssh/id_rsa --sudo --node-name web1`

That should work!

I would also suggest that the user you pass as the --ssh-user has passwordless sudo access.

3.Now we can check, knife automatically register CentOS Node2 , with Chef Server

4.Go To Node2-CentOS Terminal, Check the Home page

More on Chef

To get no.of Nodes in Chef server

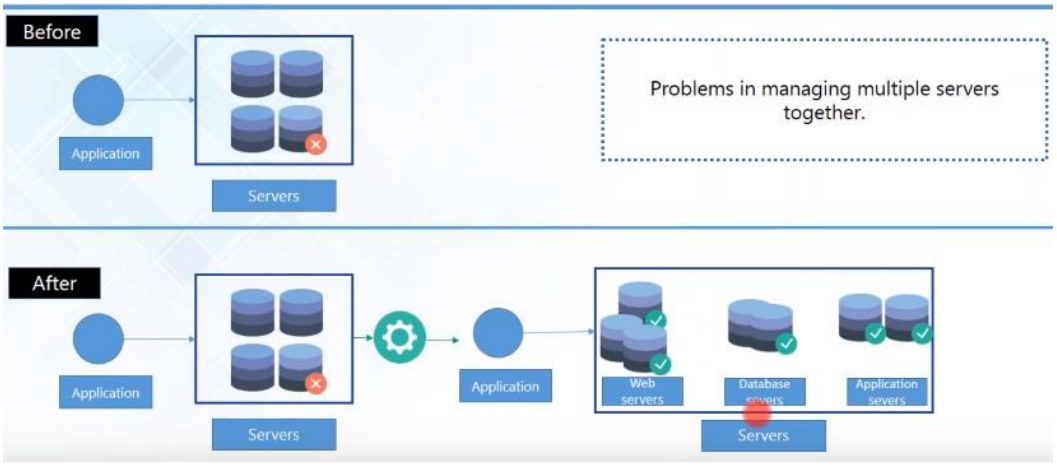
```
chef-repo> knife node list

cnode1
```

Get more Node Info

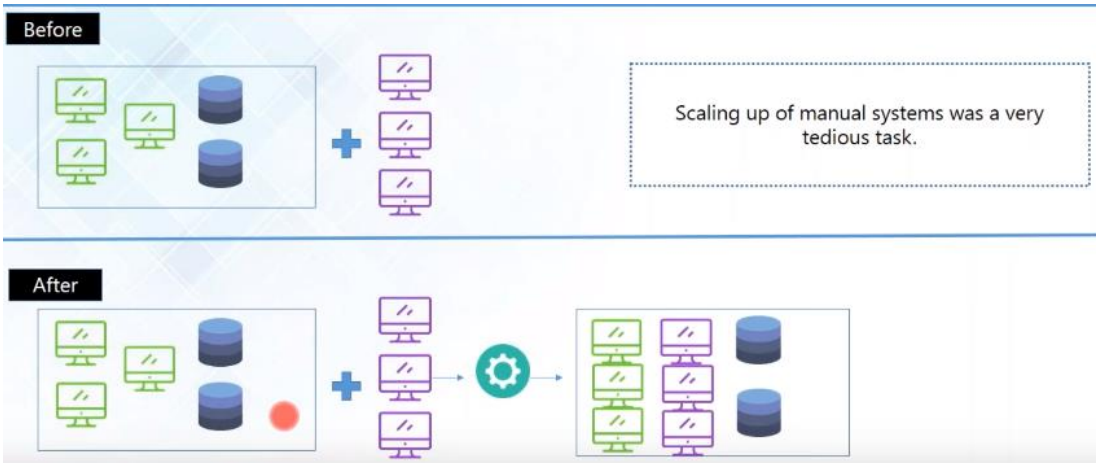
```
chef-repo> knife node show cnode1
Node Name:   cnode1
Environment: _default
FQDN:        vagrant-ubuntu-trusty-64
IP:          10.0.2.15
Run List:    recipe[learn_chef_apache2]
Roles:
Recipes:     learn_chef_apache2
Platform:    ubuntu 14.04
Tags:
```

9. Ansible

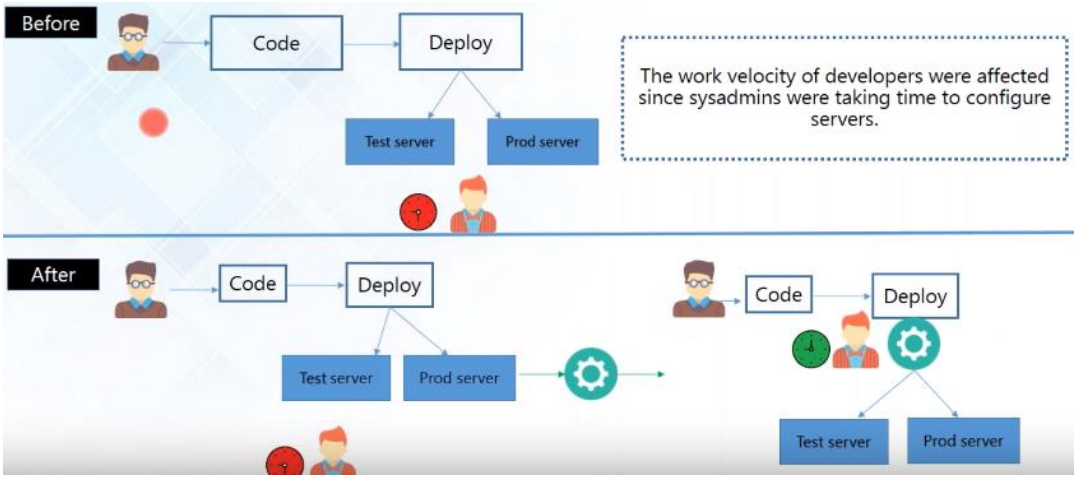
- Managing Multiple Systems/ Servers



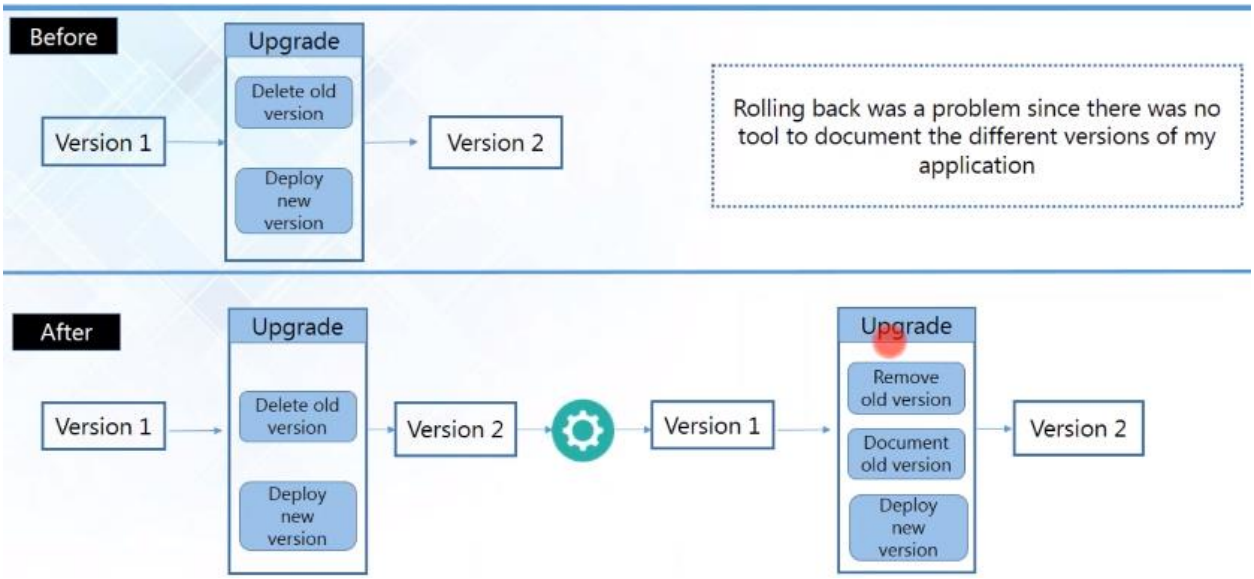
- **Scaling** -Bigbillion day, more traffic – need to add more Servers & Load Balancers



- **Time** – Configurting each server by going manually gioing into it.



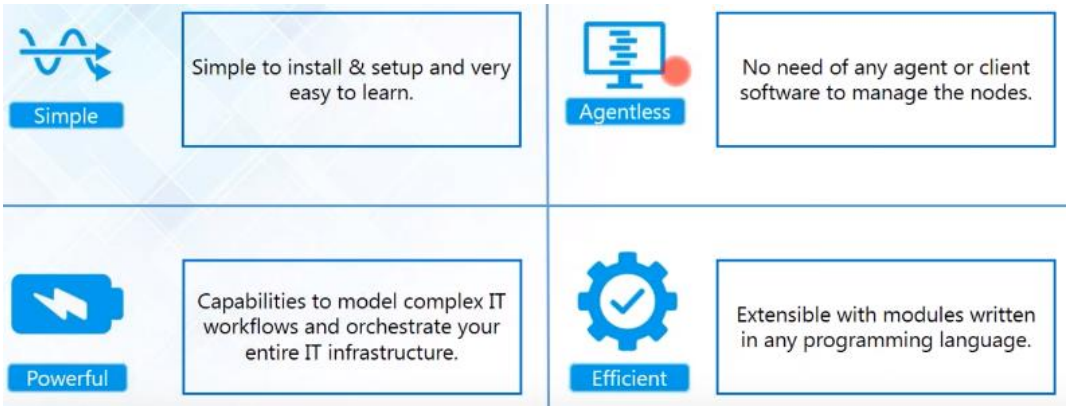
- **Rooling** – Rooling & Rooling back if Users don't like new features.



Configuration Management Tools

- **Push Based** – we can push the changes to Node machines directly (Without Serve)
- **Pull Based** – They pull all the Configurations through the Server

Features

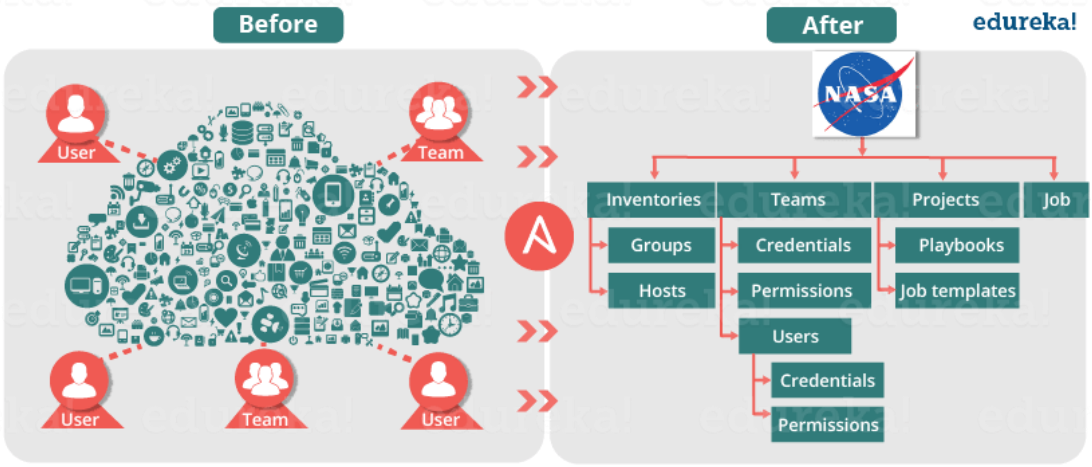


Ansible Case Study – A Real Life Usage by NASA

- NASA needed to move 65 applications from a traditional hardware based data center to a cloud-based environment for better agility and cost savings
- The rapid timeline resulted in many applications being migrated 'as it is' to a cloud environment. This created an environment which spanned multiple virtual private clouds (VPCs) and AWS accounts that could not be managed easily.
- Even simple things, like ensuring every system administrator had access to every server, or simple security patching, were extremely cumbersome.

The solution was to leverage Ansible Tower to manage and schedule the cloud environment.

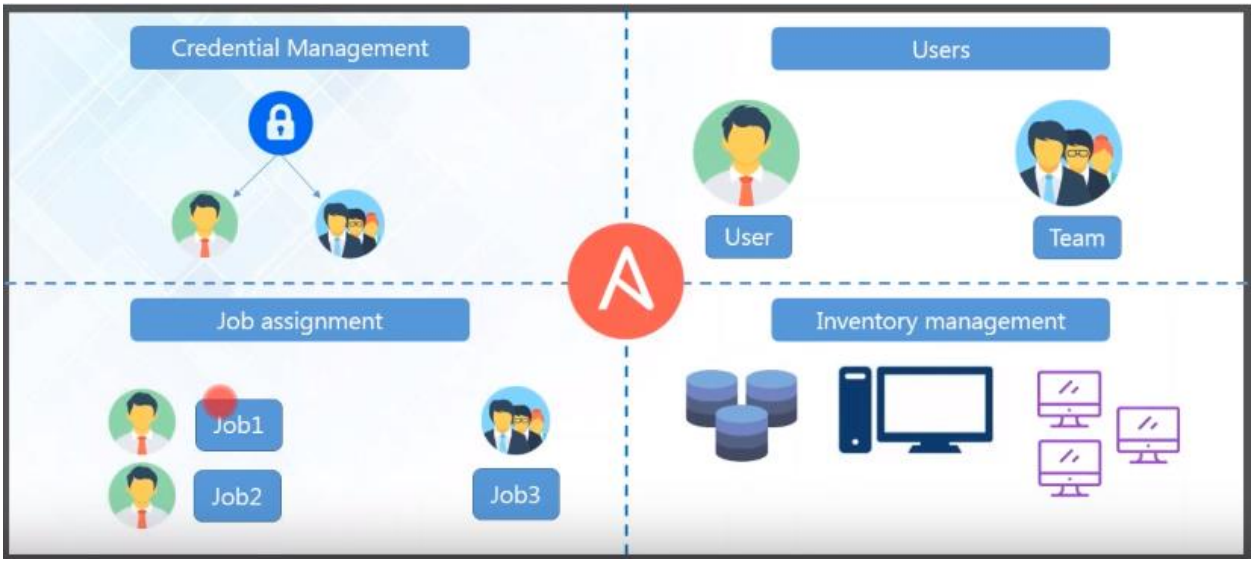
- Ansible Tower provided with a dashboard which provided the status summary of all hosts and jobs which allowed NASA to group all contents and manage access permissions across different departments
- Ansible Tower is a web-based interface for managing Ansible. One of the top items in Ansible users' wishlists was an easy-to-use UI for managing quick deployments and monitoring one's configurations.
- Further, Ansible divided the tasks among teams by assigning various roles. It managed the clean up of old job history, activity streams, data marked for deletion and system tracking info

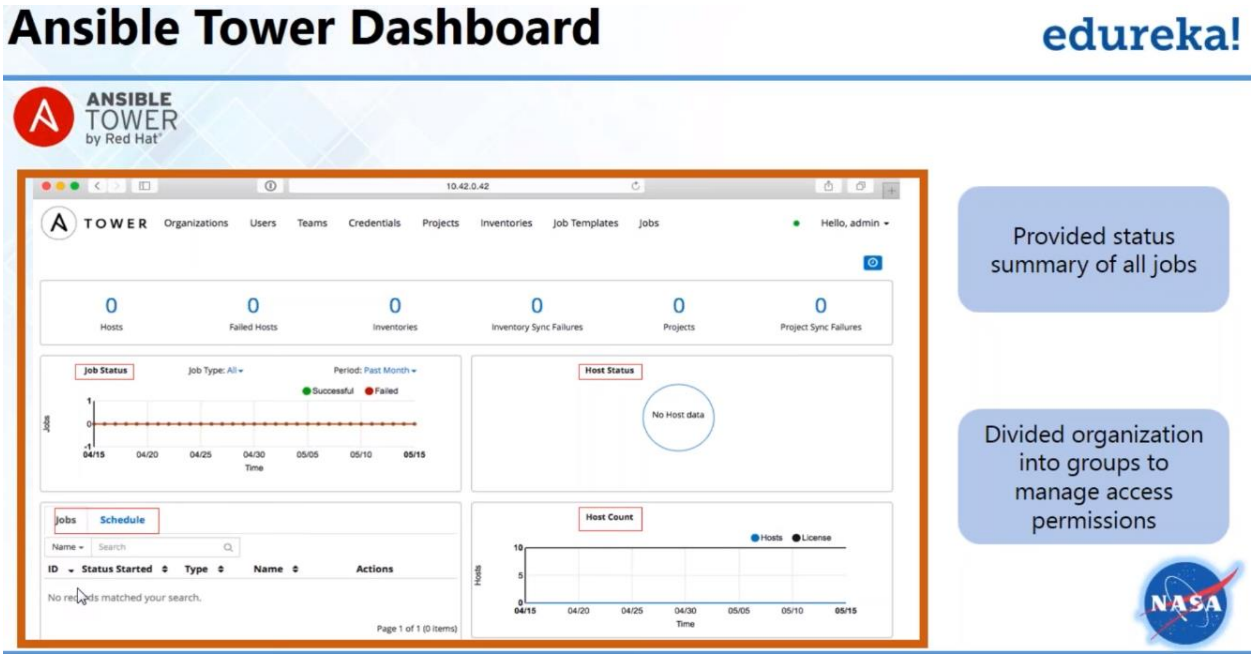


As a result, NASA has achieved the following efficiencies:

- NASA web app servers are being patched routinely and automatically through Ansible Tower with a very simple 10-line Ansible playbook.
- Every single week, both the full and mobile versions of www.nasa.gov are updated via Ansible, generally only taking about 5 minutes to do.

Ansible Tower





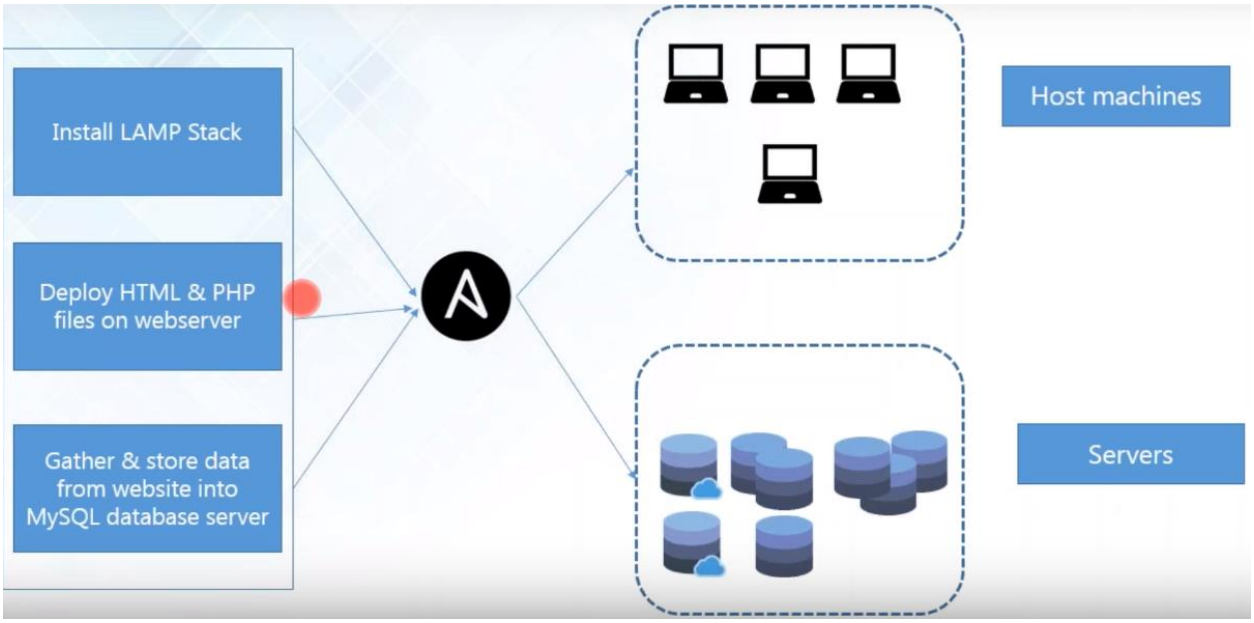
Orchestration

Orchestration – Flow of Order of Configurations.

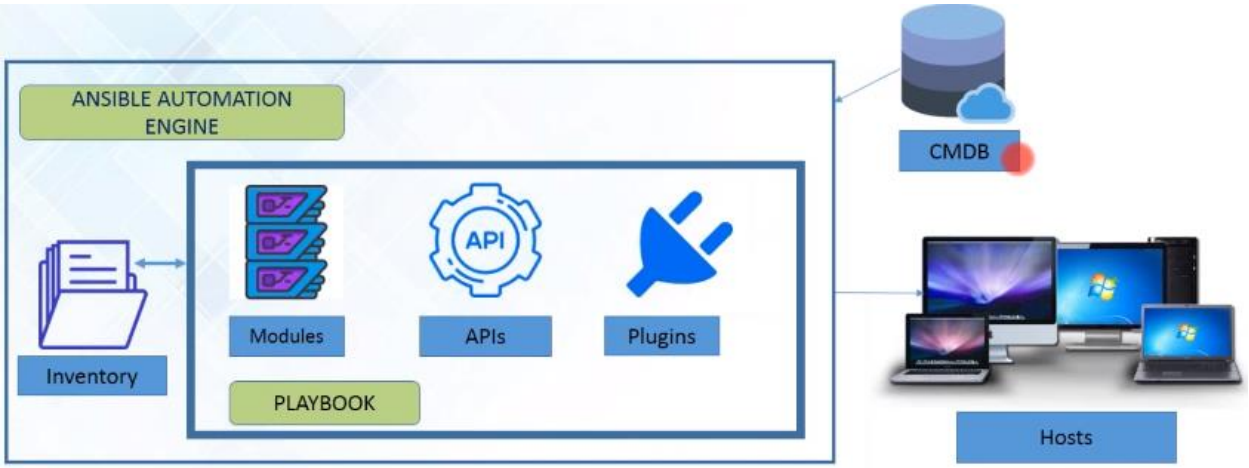
Ex: if host PHP WebApplication to Node machine

- First install LAMP Server
- Second, Deploy html & php files
- Third, get input data from WebApp & Store it to database.

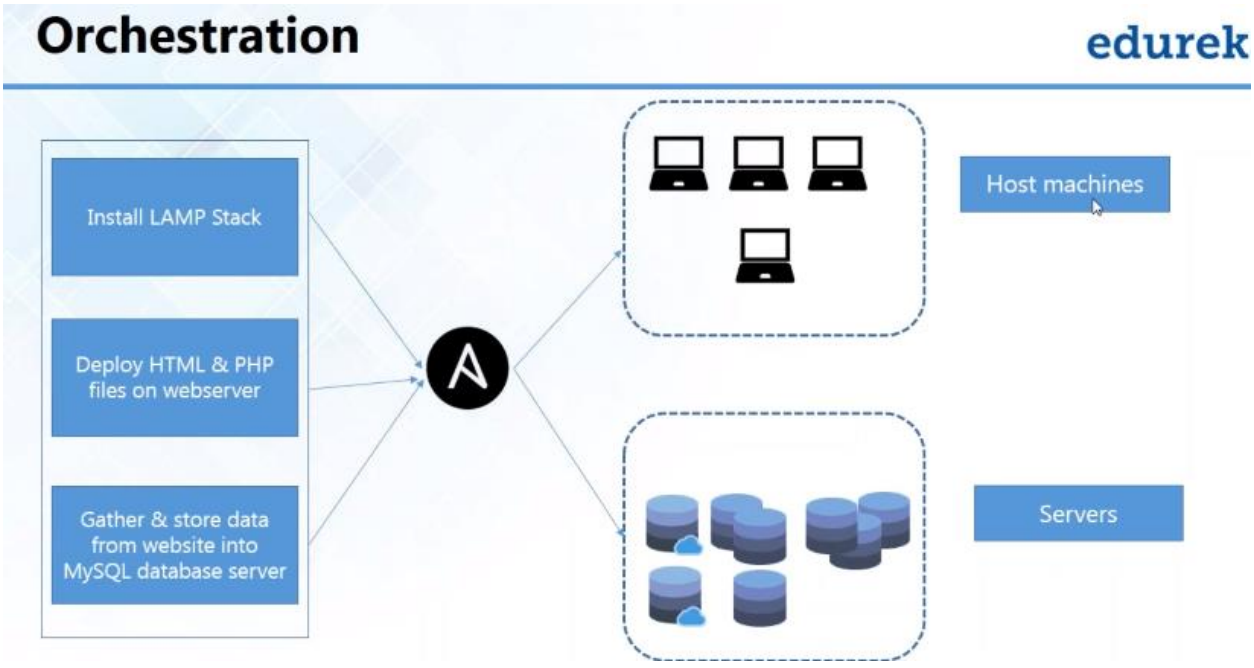
See, if we change the Order the WebApplication not Deploy properly , where you place html files with out 1st step 😊



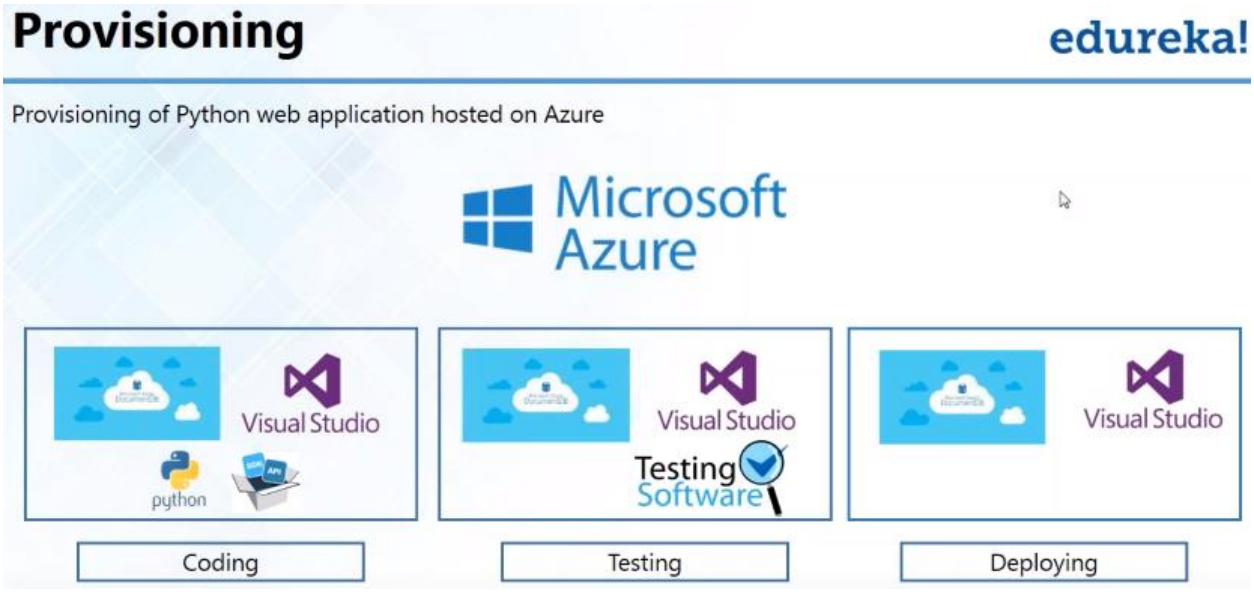
Architeture

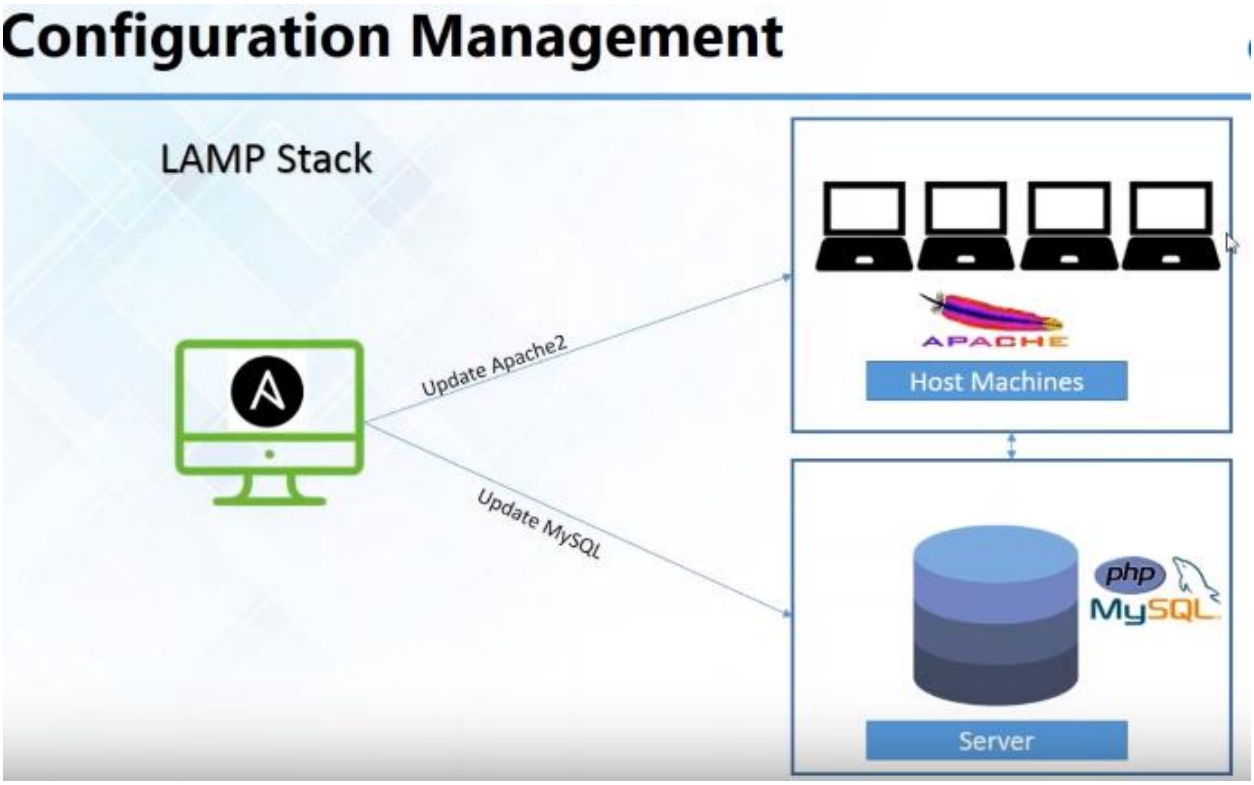


Ansible Used for



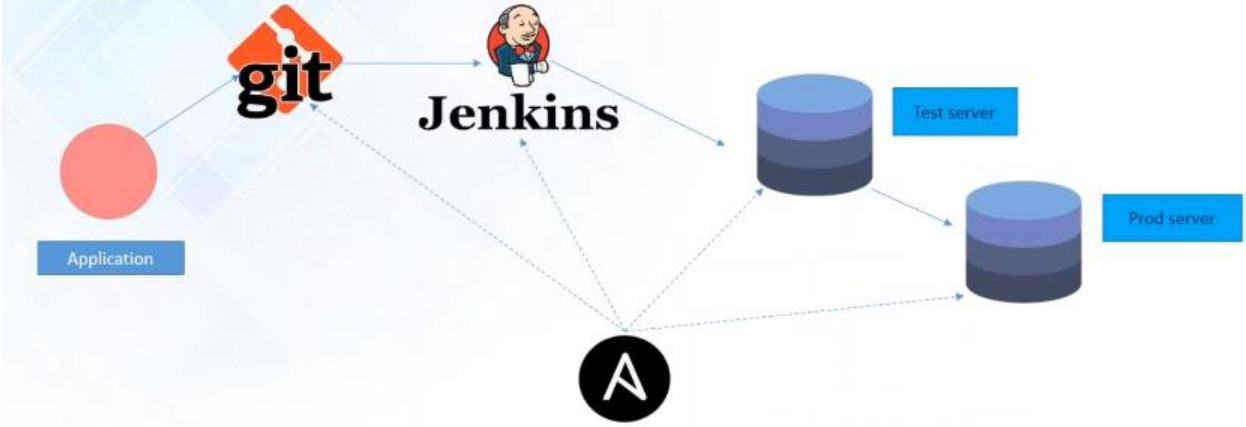
Provisioning – Installing nessasary softwares which are required to run a application properly





Application Deployment

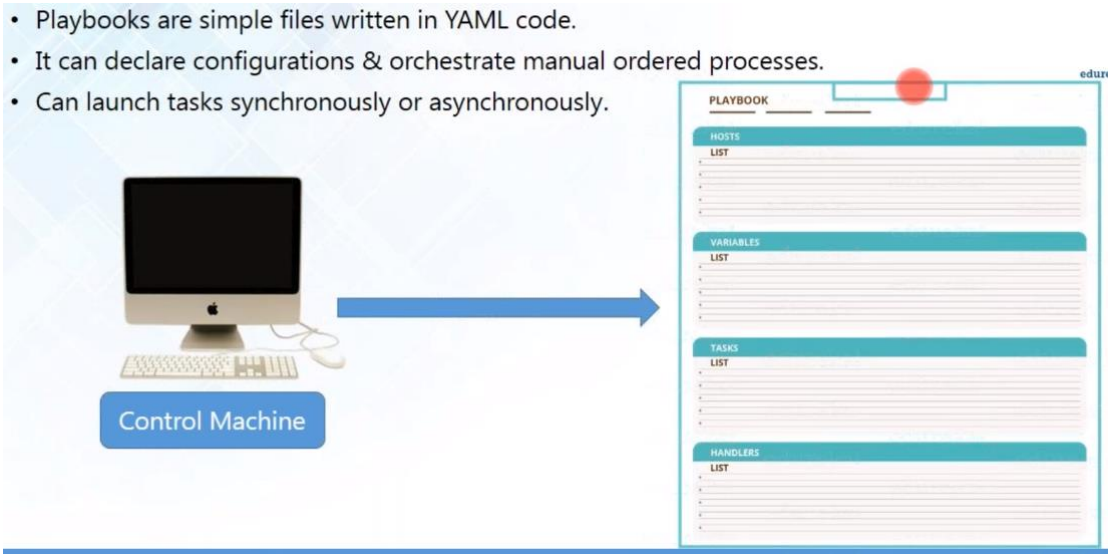
edureka!



Security & Compliance

edu

PlayBooks



Handson Task

Task 1: Installing & Configuring Control Machine

Using vagrant **(user/pwd = vagrant/vagrant)**

Control Machine (ubuntu/trusty): **10.219.19.205/24**

Host(node) Machine (ubuntu16): **10.219.19.87**

```
sudo http_proxy=http://10.219.2.220:80 apt-get update
sudo http_proxy=http://10.219.2.220:80 apt install software-properties-common
```

Then add the Ansible PPA by typing the following command:

```
sudo apt-add-repository ppa:ansible/ansible
```

Install the Ansible software

```
sudo http_proxy=http://10.219.2.220:80 apt-get install ansible
```

Check ansible version

```
ansible --version
```

Check the host inventory file, path is **/etc/ansible/hosts;** provide host machine(node) ip address in that host's file.

```
[test-servers]

10.219.19.87
```

Generate **ssh key** in the ansible machine, which we have to copy to all the remote hosts for doing deployments or configurations on them.

```
vagrant@vagrant-ubuntu-trusty-64:~/ansible$
ssh-keygen -t rsa -b 4096 -C "vagrant@vagrant-ubuntu-trusty-64"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vagrant/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vagrant/.ssh/id_rsa.
Your public key has been saved in /home/vagrant/.ssh/id_rsa.pub.
The key fingerprint is:
61:a0:f0:74:66:79:7b:99:df:af:f1:c6:54:68:3e:d6 vagrant@vagrant-ubuntu-trusty-64
The key's randomart image is:
+--[ RSA 4096]----+
|  . . =.         |
|   + =...        |
|               o+|
|              .o.|
+-----------------+
```

Ssh key is generated at (/home/vagrant/.ssh/id_rsa), here **id_rsa** is a SSH key file

Ansible – Ubuntu Hands on

Configured in Puppet Folder under ansible

https://www.edureka.co/blog/install-ansible/

https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-centos-7

Ansible Master(Control System)

Step 1 — Installing Ansible

Update Pkgs

```
[root@AnsibleMaster vagrant]# yum update -y
```

Ansible by default not available to Yum pkgs, so To get Ansible for CentOS 7, first ensure that the CentOS 7 EPEL repository is installed:EPEL(Extra Pkgs for Entraprize Linux)

```
sudo yum install epel-release
[root@AnsibleMaster vagrant]# sudo yum install epel-release
```

Once the repository is installed, install Ansible with yum:

```
sudo yum install ansible
```

CHECK Version

```
[root@AnsibleMaster vagrant]# ansible --version
ansible 2.6.4
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/root/.ansible/plugins/modules',
u'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python2.7/site-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.5 (default, Apr 11 2018, 07:36:10) [GCC 4.8.5 20150623 (Red Hat
4.8.5-28)]
[root@AnsibleMaster vagrant]#
```

Generate SSH key on the Ansible Control Machine.

```
[root@AnsibleMaster vagrant]# ssh-keygen
```

copy public key of Ansible server to its nodes.Here my node ip is : 192.168.0.108

```
ssh-copy-id -i root@<ip address of your node machine>
[root@AnsibleMaster vagrant]# ssh-copy-id -i root@192.168.0.108
```

Error : Permission denied (publickey,gssapi-keyex,gssapi-with-mic).

Go to /etc/ssh/sshd_config uncomment 'PasswordAuthentication yes' then re-started the service 'sudo systemctl restart sshd'[root@AnsibleMaster vagrant]# ssh-copy-id -i root@192.168.0.110

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any
that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now
it is to install the new keys
root@192.168.0.110's password:
Number of key(s) added: 1
```

Now try logging into the machine, with: "ssh 'root@192.168.0.110'"

and check to make sure that only the key(s) you wanted were added.

Test SSH Connection with host system

```
[root@AnsibleMaster vagrant]# ssh 'root@192.168.0.110'
[root@CentOS7-Agent ~]# who
vagrant  pts/0        2018-09-29 18:46 (10.0.2.2)
root     pts/1        2018-09-29 18:55 (192.168.0.107)
[root@CentOS7-Agent ~]#
```

Step 2 — Configuring Ansible Hosts

Ansible keeps track of all of the Nodes by reading 'hosts' file.

Just write all of our Node System details in this file

```
sudo nano /etc/ansible/hosts
```

```
[root@AnsibleMaster vagrant]# sudo vi /etc/ansible/hosts
[test-servers]
192.168.0.110
```

Step 3 — Using Simple Ansible Commands

Ping all of the node servers you configured by typing:

```
[root@AnsibleMaster vagrant]# ansible -m ping all
192.168.0.110 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Check uptime of your node machines

```
[root@AnsibleMaster vagrant]# ansible -m command -a "uptime" 'test-servers'
192.168.0.110 | SUCCESS | rc=0 >>
 19:06:37 up 21 min,  2 users,  load average: 0.00, 0.02, 0.05
```

Check kernel version of your nodes

```
[root@AnsibleMaster vagrant]# ansible -m command -a "uname" 'test-servers'
192.168.0.110 | SUCCESS | rc=0 >>
Linux
```

Step 4 : Hands on : PlayBook to Deploy Nginx Using Ansible

Nginx is software to provide a web server. It can act as a reverse proxy server for TCP, UDP, HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a load balancer and an HTTP cache.

Write Play Book : nginx.yml

```
---
-
  become: true
  hosts: test-servers
  name: "Install nginx"
  tasks:
    -
      name: "Add epel-release repo"
      yum:
        name: epel-release
        state: present
    -
      name: "Install nginx"
      yum:
        name: nginx
        state: present
    -
      name: "Start NGiNX"
      service:
```

Run the playbook, it will install nginx on nodes

```
[root@AnsibleMaster vagrant]# ansible-playbook nginx.yml
```

```
PLAY [Install nginx]
*********************************************************************************
*****************

TASK [Gathering Facts]
*********************************************************************************
***************
ok: [192.168.0.110]

TASK [Add epel-release repo]
*********************************************************************************
*********
ok: [192.168.0.110]

TASK [Install nginx]
*********************************************************************************
*****************
ok: [192.168.0.110]

TASK [Start NGiNX]
*********************************************************************************
*******************
changed: [192.168.0.110]

PLAY RECAP
*********************************************************************************
**************************
192.168.0.110              : ok=4    changed=1    unreachable=0    failed=0
```

Now to check if it is installed in your node machine, type the following command in your node:

```
ps waux | grep nginx
[vagrant@CentOS7-Agent ~]$ ps waux | grep nginx
root      5600  0.0  0.4 120812  2096 ?        Ss   19:53   0:00 nginx: master process
/usr/sbin/nginx
nginx     5601  0.0  0.6 121276  3132 ?        S    19:53   0:00 nginx: worker process
vagrant   5626  0.0  0.1  12520   952 pts/0    S+   19:55   0:00 grep --color=auto
nginx
```

Ansible Master on AWS Cloud

https://www.youtube.com/watch?v=wpIgvy34BzU

https://www.edureka.co/blog/aws-devops-a-new-approach-to-software-deployment/

1.Login to Aws

2.LaunchInstance > Red Hat Enterprise Linux 7.5 > t2.micro

Instances : 4

Auto-assign Public IP : Enable

Tag : name = Node

Configure Security Group

Security group :AnsibleSecureGroup

Download public key "AnsibleNodes.pem"and launch the nodes

3.Assign "Elastic IPs"

Public IP will keep change, if we reboot/re-launch the System. To make IP Address Static we need assign "Elasitic IP"

-Newwork secirity tab > Elastic Ips > Allocate 4 New IP address ::scope -vpc

-Associte Elastic IPS with Nodes

-Select IP > Actions > Associte :: [tick] Re-Associate

- Now AWS releses the Public IP & makes E'ip as public Ip

Addresses:13.126.179.188, 13.127.176.9, 13.232.223.255, 13.233.62.166

Public IP : is for connecting with public systems over internet

Private IP: is for connecting Internal System resorces. ex: apache runs on port 8080, mysql:3305

4.Connect with Ansible Master & Nodes

Locate "AnsibleNodes.pem" by terminal & chane permissions before connect

```
chmod 400 AnsibleNodes.pem

ssh -i "AnsibleNodes.pem" ec2-user@ec2-13-126-179-188.ap-south-1.compute.amazonaws.com
```

```
satya@satya:~/.../ansible$  chmod 400 AnsibleNodes.pem
satya@satya:~/.../ansible$ ssh -i "AnsibleNodes.pem" ec2-user@ec2-13-126-179-188.ap-
south-1.compute.amazonaws.com
[ec2-user@ip-172-31-23-104 ~]$ ========== CONNECTED ================
```

Switch to Root

```
[ec2-user@ip-172-31-23-104 ~]$ sudo su
```

Connet withh nodes

```
ssh -i "AnsibleNodes.pem" ec2-user@ec2-13-127-176-9.ap-south-1.compute.amazonaws.com
ssh -i "AnsibleNodes.pem" ec2-user@ec2-13-232-223-255.ap-south-1.compute.amazonaws.com
ssh -i "AnsibleNodes.pem" ec2-user@ec2-13-233-62-166.ap-south-1.compute.amazonaws.com
```

5. Create user called "test" in master & Nodes with pwd test

```
useradd test
passwd test
```

Provide ROOT access to test user by going vi /etc/sudoers add below line

test ALL=(ALL) NOPASSWD: ALL

6.Go to sudo vi /etc/ssh/sshd_config in all nodes, enable

```
PasswordAuthentication yes
PermitRootLogin yes
```

Restart System SErvices

```
systemctl restart sshd
```

7.Now Login using Test Account in all Servres

```
ssh test@ipaddress

ssh test@13.126.179.188
ssh test@13.127.176.9
ssh test@13.232.223.255
ssh test@13.233.62.166
```

8.Get Public & private Ips

```
NODE    PUBLIC         Private
---------------------------------
Master  13.126.179.188 172.31.23.104
Node1   13.127.176.9   172.31.20.225
Node2   13.232.223.255 172.31.31.116
Node3   13.233.62.166  172.31.29.193
```

9.All the Systems are Internal to AWS. So Check login from On Node Terminal to Other For this we need to add ICMP - All to Security Group in AWS

```
ssh test@172.31.23.104

ssh test@172.31.20.225
ssh test@172.31.31.116
ssh test@172.31.29.193

[test@ip-172-31-29-193 ~]$ ssh test@172.31.20.225
test@172.31.20.225's password:
```

Last login: Sun Sep 30 14:43:23 2018 from ip-172-31-29-193.ap-south-1.compute.internal

See it is aking for Password for Internal Communication. To make Systems connect with out ask password We should generate SSH Key & Share to the Nodes

10.Generate SSH key

Login to mastet using test@publ ip & generate key

```
[test@ip-172-31-20-225 ~]$ ssh-keygen
```

Similarly Generate the Key in all the machines

-Copy Master SSH key to All the Nodes

```
 ssh-copy-id <Private-Ip>

ssh-copy-id 172.31.20.225
ssh-copy-id 172.31.31.116
ssh-copy-id 172.31.29.193
```

It will show following message in all nodes, by successfull copy

```
Number of key(s) added: 1
```

Now try logging into the machine, with: "ssh '172.31.20.225'" and check to make sure that only the key(s) you wanted were added.

-Now Copy Node1 SSH to all Other Master, 2 Nodes Machines same way

```
ssh-copy-id 172.31.23.104
```

```
ssh-copy-id 172.31.31.116
ssh-copy-id 172.31.29.193
```

-Now Copy Node2 SSH to all Other Master, 2 Nodes Machines same way

```
ssh-copy-id 172.31.23.104
ssh-copy-id 172.31.20.225
ssh-copy-id 172.31.29.193
```

Now Copy Node3 SSH to all Other Master, 2 Nodes Machines same way

```
ssh-copy-id 172.31.23.104
ssh-copy-id 172.31.20.225
ssh-copy-id 172.31.31.116
```

11.Installing Ansible

Ansibe is not free, Ansible.com provide Ansible Tower only that too Commercial.we can use Opensource version from fedora repo.

Ansible by default not available to Yum pkgs, so To get Ansible for CentOS 7, first ensure that the Redhat 7 EPEL repository is installed:EPEL(Extra Pkgs for Entraprize Linux)

```
sudo yum install epel-release
```

-Update Pkgs

```
[root@AnsibleMaster vagrant]# yum update -y
[root@AnsibleMaster vagrant]# sudo yum install epel-release
```

-Add Fedora repo to yum

```
sudo yum install wget
wget https://archive.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
sudo rpm -ivh epel-release-latest-7.noarch.rpm
sudo yum update -y
```

-You can epel repo is added to our system

```
[test@ip-172-31-29-193 ~]$ ls -la /etc/yum.repos.d/
total 40
drwxr-xr-x.  2 root root  142 Sep 30 16:49 .
drwxr-xr-x. 75 root root 8192 Sep 30 16:35 ..
-rw-r--r--.  1 root root  951 Oct  2  2017 epel.repo
-rw-r--r--.  1 root root 1050 Oct  2  2017 epel-testing.repo
```

```
-rw-r--r--.  1 root root  607 Sep 30 08:58 redhat-rhui-client-config.repo
-rw-r--r--.  1 root root 8679 Sep 30 08:58 redhat-rhui.repo
-rw-r--r--.  1 root root   82 Sep 30 08:58 rhui-load-balancers.conf
```

-Once the repository is installed, install Ansible with yum:

```
sudo yum install ansible -y
```

Step 2 — Configuring Ansible Hosts

Ansible keeps track of all of the Nodes by reading 'hosts' file.

Just write all of our Node System details in this file

```
sudo vi /etc/ansible/hosts
[test-servers]
172.31.20.225
172.31.31.116
172.31.29.193
```

13. Using Simple Ansible Commands

Ping all of the node servers you configured by typing:

```
[test@ip-172-31-31-116 ~]$ ansible -m ping all
172.31.29.193 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
172.31.31.116 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
172.31.20.225 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Check uptime of your node machines

```
172.31.29.193 | SUCCESS | rc=0 >>
 17:25:28 up  8:27,  2 users,  load average: 0.00, 1.25, 11.17

172.31.31.116 | SUCCESS | rc=0 >>
 17:25:28 up  8:27,  2 users,  load average: 0.02, 0.02, 0.05

172.31.20.225 | SUCCESS | rc=0 >>
 17:25:29 up  8:27,  2 users,  load average: 1.31, 1.16, 1.17
```

14 : Hands on : PlayBook to Deploy Nginx Using Ansible

Nginx is software to provide a web server. It can act as a reverse proxy server for TCP, UDP, HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a load balancer and an HTTP cache.

```
vi playbook.yml
```

Write Play Book : playbook.yml

```
---
-
  become: true
  hosts: test-servers
  name: "Install nginx"
  tasks:
    -
      name: "Add epel-release repo"
      yum:
        name: epel-release
        state: present
    -
      name: "Install nginx"
      yum:
        name: nginx
        state: present
    -
      name: "Start NGiNX"
      service:
```

Run the playbook, it will install nginx on nodes

```
[root@AnsibleMaster vagrant]# ansible-playbook playbook.yml

[test@Kira ~]$ ansible-playbook playbook.yml
```

```
PLAY [Install nginx]
*********************************************************************************
*****************
```

```
TASK [Gathering Facts]
*********************************************************************************
***************
ok: [172.31.31.116]
ok: [172.31.29.193]
ok: [172.31.20.225]

TASK [Add epel-release repo]
*********************************************************************************
*********
ok: [172.31.29.193]
ok: [172.31.31.116]
ok: [172.31.20.225]

TASK [Install nginx]
*********************************************************************************
*****************
ok: [172.31.29.193]
ok: [172.31.31.116]
ok: [172.31.20.225]

TASK [Start NGiNX]
*********************************************************************************
*******************
ok: [172.31.31.116]
ok: [172.31.29.193]
ok: [172.31.20.225]

PLAY RECAP
*********************************************************************************
**************************
172.31.20.225              : ok=4    changed=0    unreachable=0    failed=0
172.31.29.193              : ok=4    changed=0    unreachable=0    failed=0
172.31.31.116              : ok=4    changed=0    unreachable=0    failed=0
```

Now to check if it is installed in your node machine, type the following command in your node:

```
ps waux | grep nginx
```

References

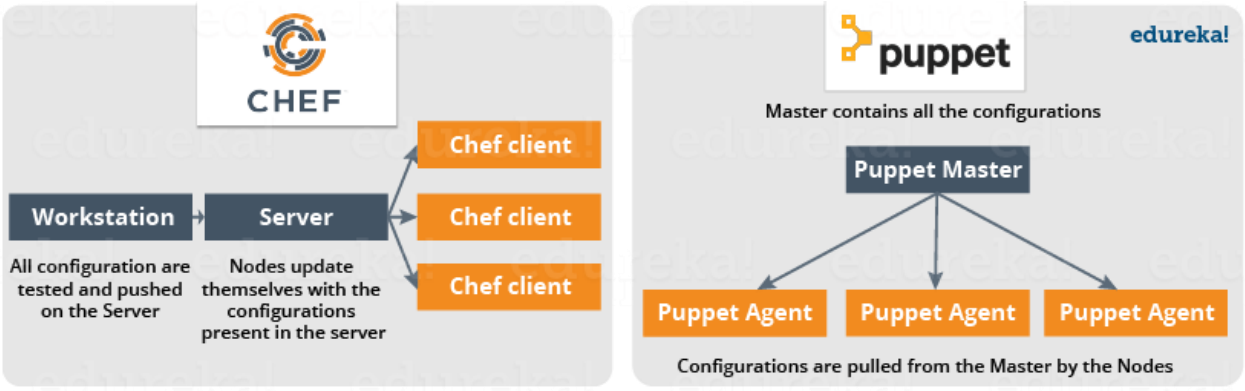
Ansible Install : https://youtu.be/XJpN8qpxWbA

Ansible Woman : https://youtu.be/dCQpaTTTv98

AWS + Ansible : https://youtu.be/wpIgvy34BzU

10. Puppet

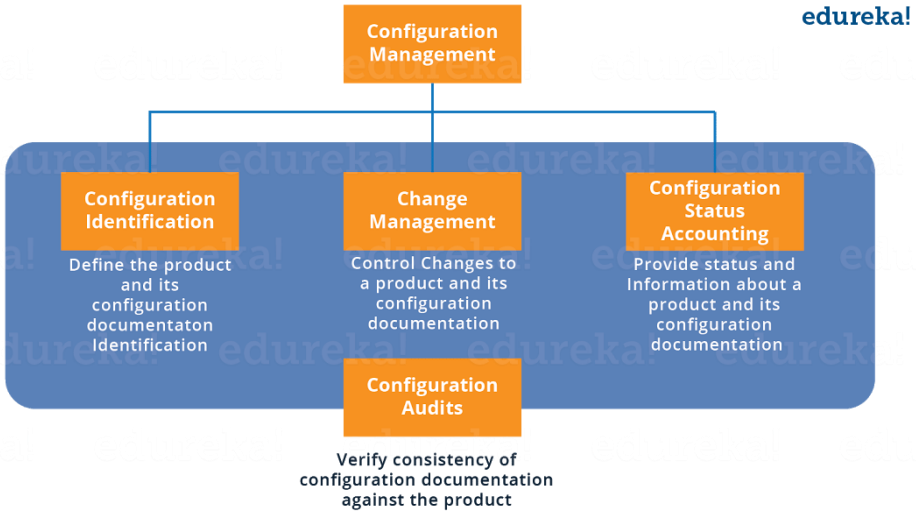
Puppet : a person, group, or country under the control of another.



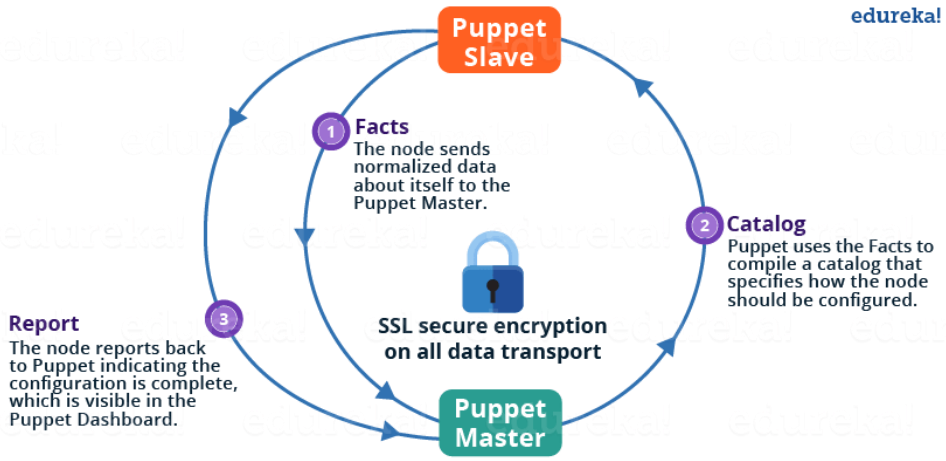
What Is Puppet?

Puppet is a Configuration Management tool that is used for deploying, configuring and managing servers. It performs the following functions:

- Defining distinct configurations for each and every host, and continuously checking and confirming whether the required configuration is in place and is not altered (if altered Puppet will revert back to the required configuration) on the host.
- Dynamic scaling-up and scaling-down of machines.
- Providing control over all your configured machines, so a centralized (master-server or repo-based) change gets propagated to all, automatically.



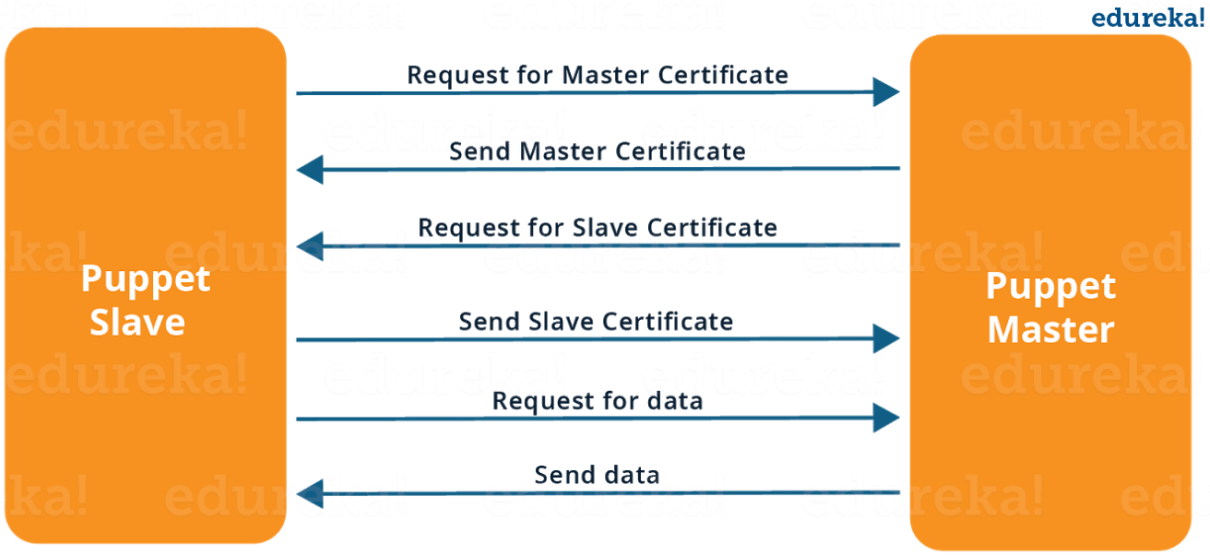
Architecture of Puppet



The following functions are performed in the above image:

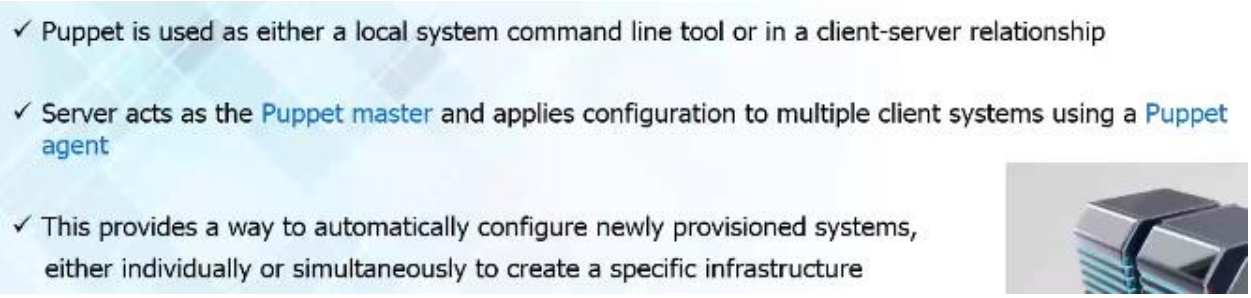
- The Puppet Agent sends the Facts to the Puppet Master. Facts are basically key/value data pair that represents some aspect of Slave state, such as its IP address, up-time, operating system, or whether it's a virtual machine. I will explain Facts in detail later in the blog.
- Puppet Master uses the facts to compile a Catalog that defines how the Slave should be configured. Catalog is a document that describes the desired state for each resource that Puppet Master manages on a Slave. I will explain catalogs and resources in detail later.
- Puppet Slave reports back to Master indicating that Configuration is complete, which is visible in the Puppet dashboard.

Puppet Master and Slave Communication



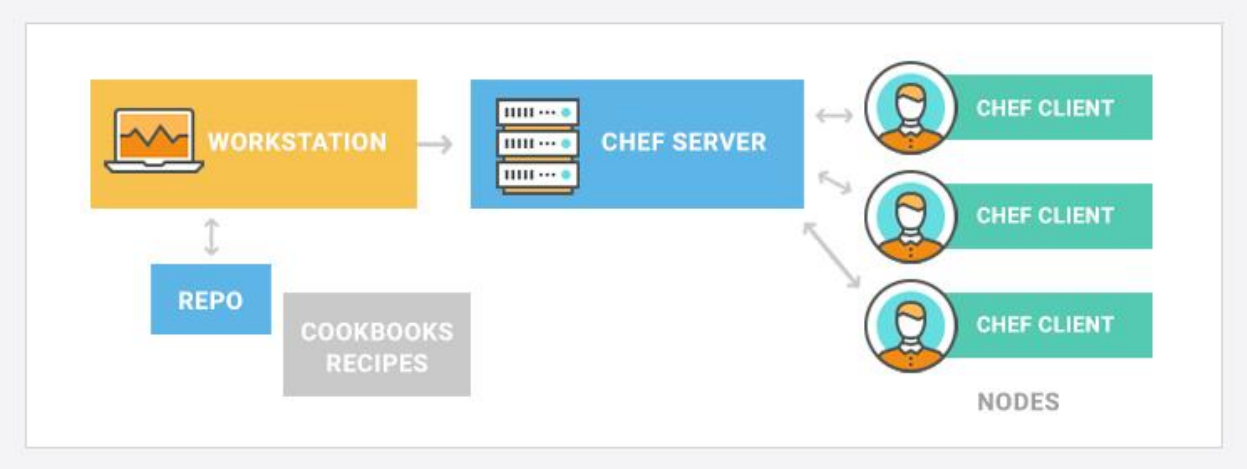
As you can see from the above Image:

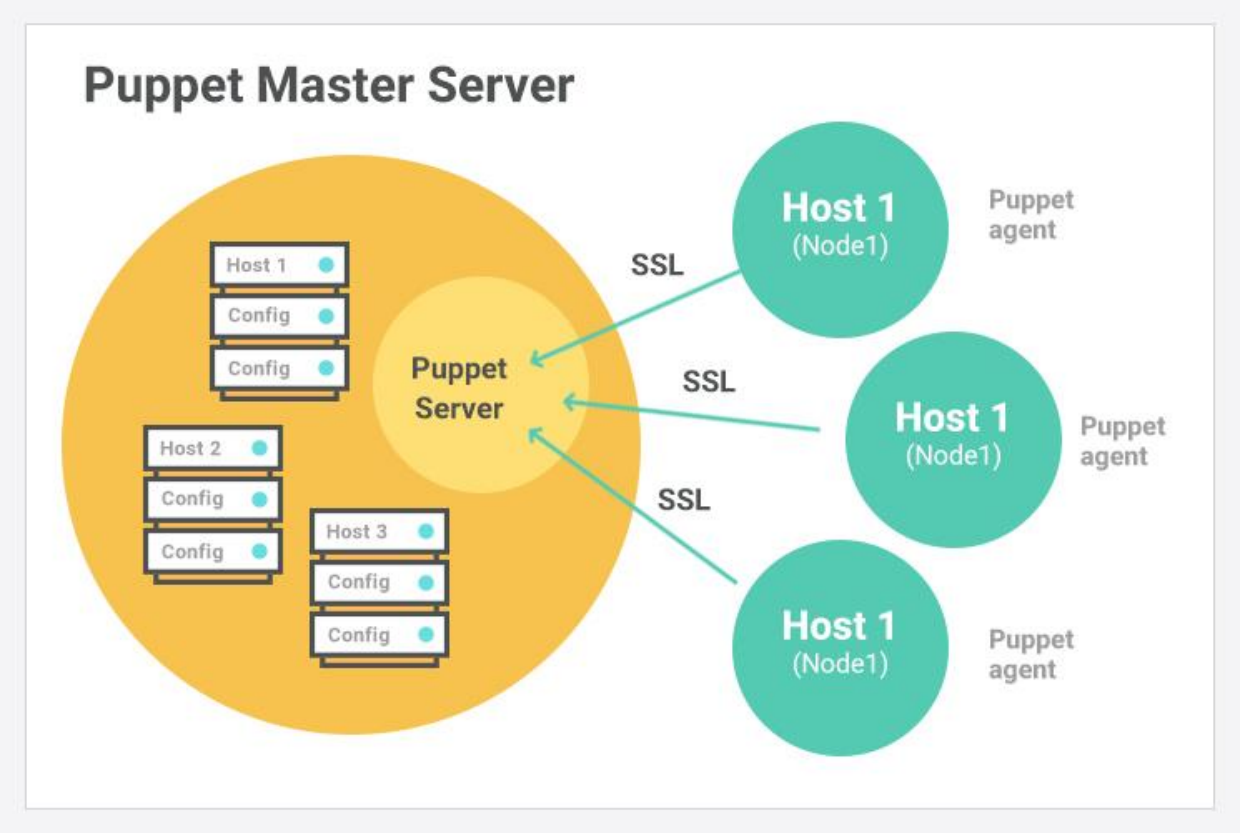
- Puppet Slave asks for Puppet Master certificate.
- After receiving Puppet Master certificate, Master requests for Slave certificate.
- Once Master has signed the Slave certificate, Slave requests for configuration/data.
- Finally, Puppet Master will send the configuration to Puppet Slave.



Rudy based Tool

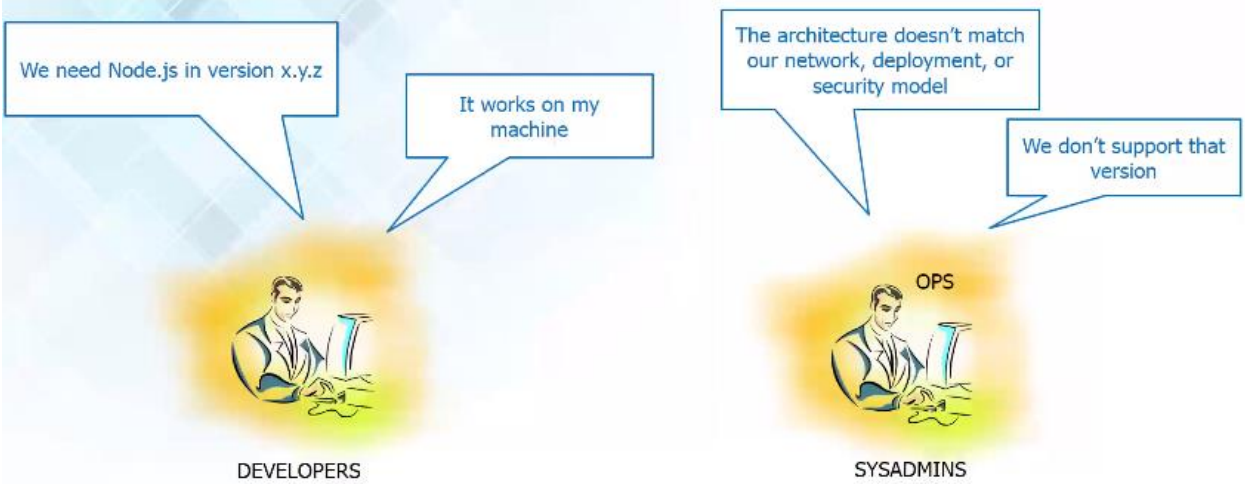
Chef vs Puppet





https://logz.io/blog/chef-vs-puppet/

What the Problems it Solves



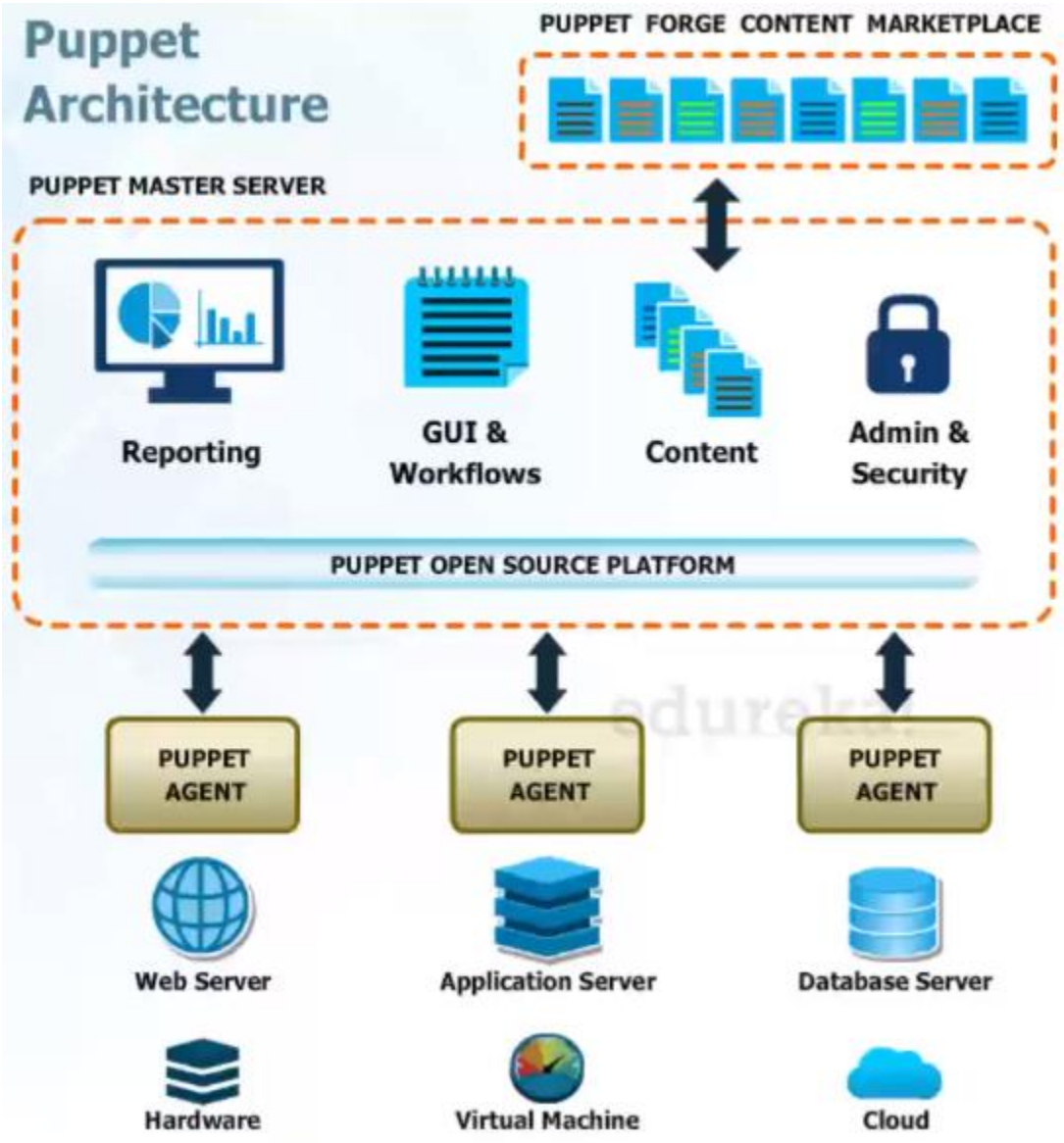
- ✓ Infrastructure as Code:
  - Track
  - Test
  - Deploy
  - Reproduce
  - Scale
- ✓ Scale: You could bring new servers online in minutes because they can be setup automatically by your Configuration Management system?
- ✓ Code commit log shows the history of change on the infrastructure
- ✓ Reproducible setups: Do once, repeat forever
- ✓ Scale quickly: Done for one, use on many
- ✓ Coherent and consistent server setups
- ✓ Aligned Environments for Development, Test, QA, Prodution nodes
- ✓ Alternatives to Puppet: Chef, CFEngine, Salt, Ansible

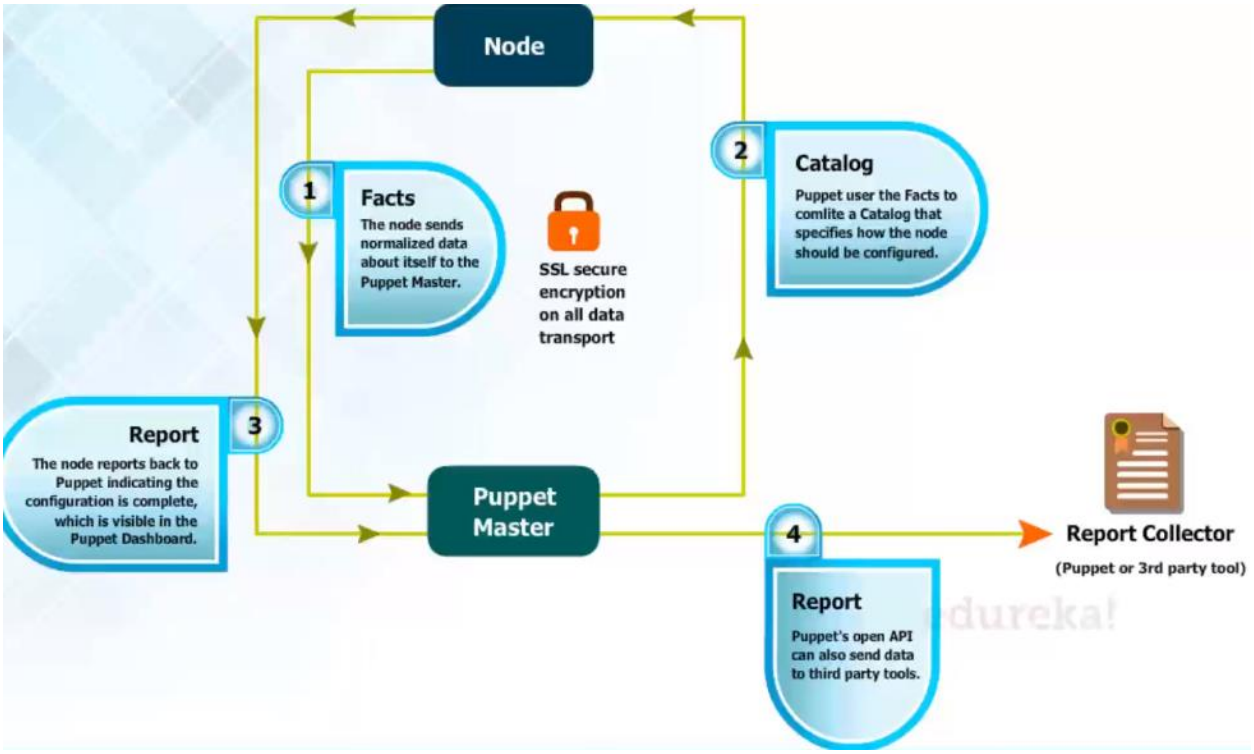
The Puppet Environment

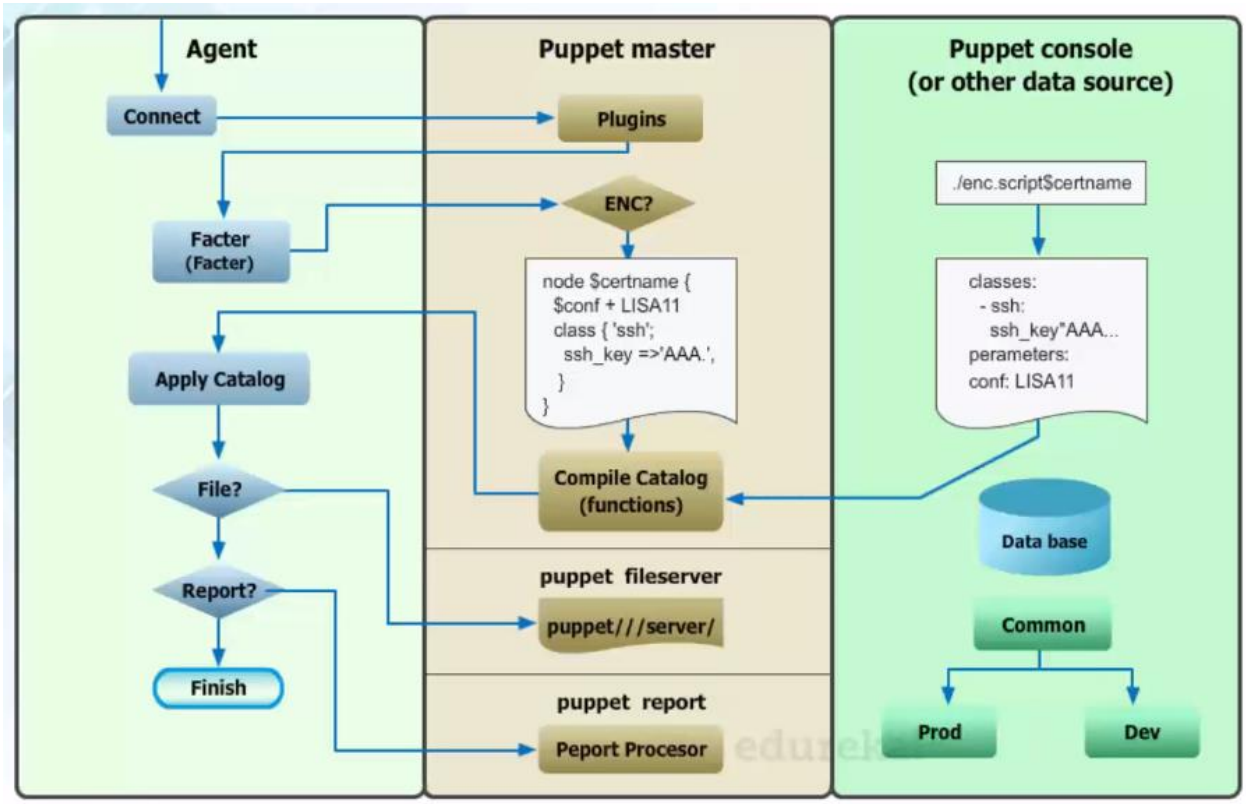
- ✓ Puppet Labs- The Company behind Puppet
- ✓ Puppet- The OpenSource version
- ✓ Puppet Enterprise - The commercial version
- ✓ Puppet Documentation - Main and Official reference
- ✓ Puppet Modules on: Module Forge and GitHub
- ✓ The Community – Forums, User groups etc.

Software related to Puppet:

- ✓ Facter - is Puppet's cross-platform system profiling library. It finds and reports per-node facts, which are available in Puppet manifests as variables.(Complementary tool to retrieve system's data)
- ✓ MCollective - Infrastructure Orchestration framework
- ✓ Hiera - Puppet data can be placed using this Key-value lookup tool
- ✓ PuppetDB - Data generated by Puppet is stored here
- ✓ Puppet DashBoard - A Puppet Web frontend and External Node Classifier (ENC)
- ✓ The Foreman - A well-known third party provisioning tool and Puppet ENC
- ✓ Geppetto - A Puppet IDE based on Eclipse

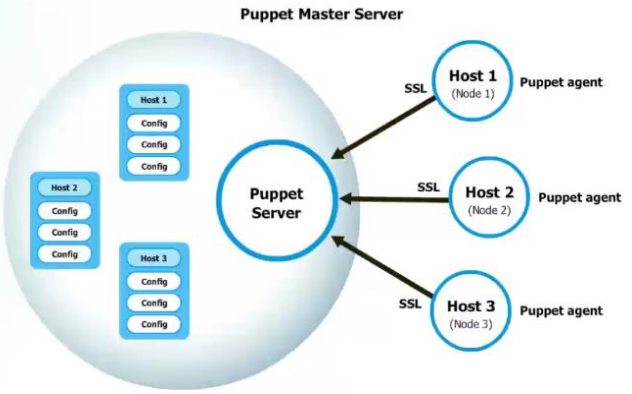


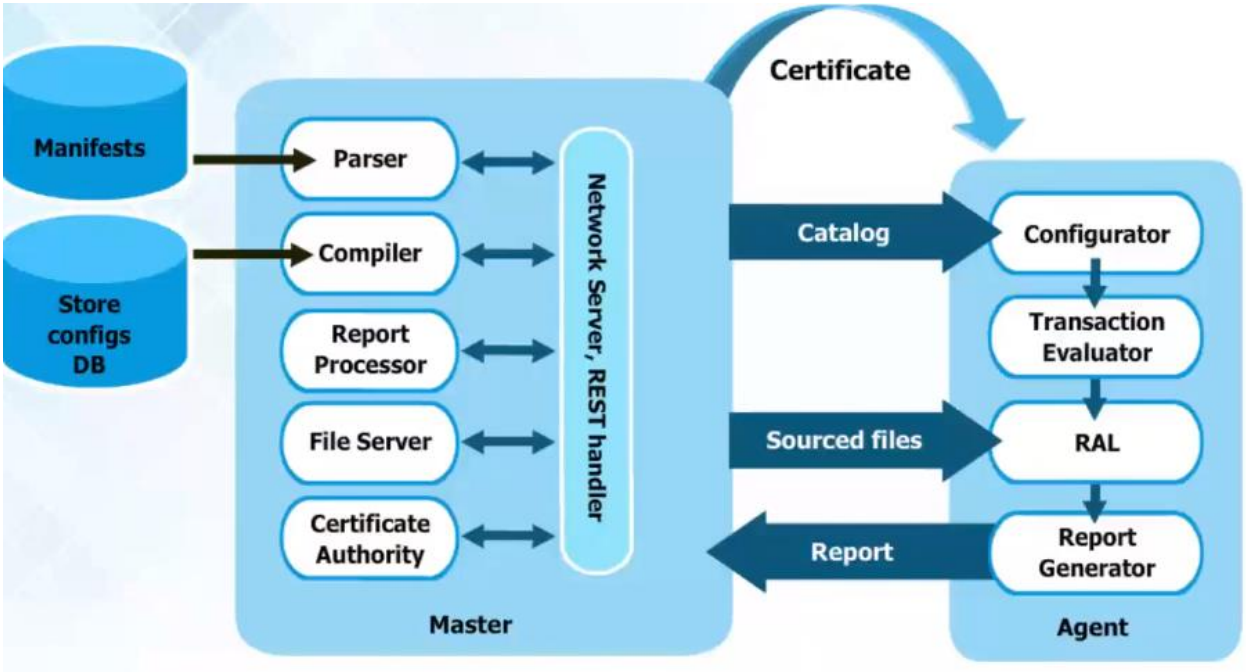

Node
1
Facts
The node sends normalized data about itself to the Puppet Master.
SSL secure encryption on all data transport
2
Catalog
Puppet user the Facts to comlite a Catalog that specifies how the node should be configured.
3
Report
The node reports back to Puppet indicating the configuration is complete, which is visible in the Puppet Dashboard.
Puppet Master
4
Report
Puppet's open API can also send data to third party tools.
Report Collector
(Puppet or 3rd party tool)


Agent
Connect
Facter (Facter)
Apply Catalog
File?
Report?
Finish
Puppet master
Plugins
ENC?
node $certname {
$conf + LISA11
class { 'ssh';
ssh_key =>'AAA.',
}
}
Compile Catalog (functions)
puppet fileserver
puppet///server/
puppet report
Peport Procesor
Puppet console (or other data source)
./enc.script$certname
classes:
- ssh:
ssh_key"AAA...
perameters:
conf: LISA11
Data base
Common
Prod
Dev

Puppet works on the Master Server model

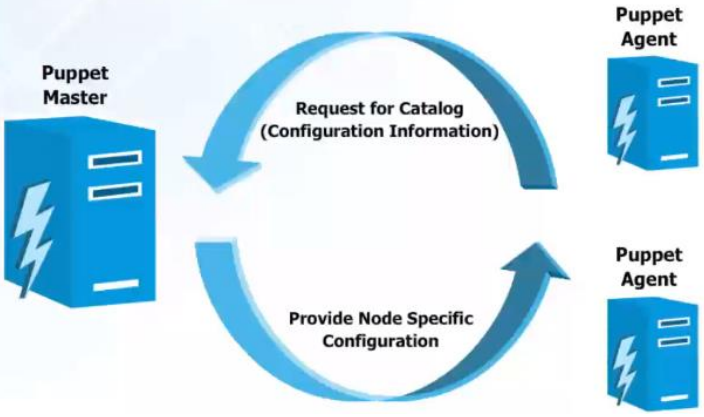
- ✓ Puppet Master: This machine contains all the configuration for different hosts. Puppet master will run as a daemon on this master server
- ✓ Puppet Agent: This is daemon which runs on each node and talks to the master
- ✓ The connection between these two machines is made in a secure encrypted channel with the help of SSL





✓ Following are the steps for Puppet configuration:

- Clients connect to the master and master identifies the configuration according to the client
- Master builds the Configuration that needs to be applied to a host, compiles it and makes it ready
- Clients pull the Configuration and apply them on the respective nodes



Puppet Terminology

Resources

- ✓ Puppet code is primarily composed of resource declarations
- ✓ A resource describes about the state of the system, e.g. a certain user or file should exist, or a package should be installed
- ✓ Resource declarations can be formatted as follows:

```
resource_type
 { 'resource_name'
  attribute => value
  ...
  }
```

- ✓ Example for Resource declaration :

```
user
{ 'mitchell':
  ensure    => present,
  uid       => '1000',
  gid       => '1000',
  shell     => '/bin/bash',
  home      => '/home/mitchell'
}
```

This resource declaration describes a user resource named 'mitchell', with the specified attributes

- ✓ To list all of the default resource types that are available to Puppet, enter the following command:

```
puppet resource --types
```

Manifests

- ✓ Puppet programs are called manifests
- ✓ Manifests are composed of puppet code and their filenames use the .pp extension
- ✓ The default main manifest in Puppet installed via apt is :

/etc/puppet/manifests/site.pp.

Class

→ In Puppet, classes are code blocks that can be called in a code elsewhere. Using classes allows you to reuse Puppet code, and can make reading manifests easier

Class Definition

```
class example_class {
  ...
  code
  ...
}
```

- ✓ Class declaration occurs when a class is called in a manifest
- ✓ A class declaration tells Puppet to evaluate the code within the class
- ✓ Class declarations come in two different flavors: normal and resource-like

A normal class declaration occurs when the include keyword is used in Puppet code, like:

```
include example_class
```

A resource-like class declaration occurs when a class is declared like a resource, like:

```
class { 'example_class': }
```

Modules

- ✓ A module is a collection of manifests and data (such as facts, files, and templates), and they have a specific directory structure
- ✓ Modules are useful for organizing your Puppet code, because they allow you to split your code into multiple manifests
- ✓ It is considered best practice to use modules to organize almost all of your Puppet manifests
- ✓ To add a module to Puppet, place it in the /etc/puppet/modules directory

TERMINOLOGY		
class	data type	defined resource
environment	exported resource	facter
function	Global scope	Hiera
host	idempotent	Inheritance(class)
Main manifest	manifest	Manifest ordering
master	masterless	module
node	node scope	Node state
parameter	Profile	Property(custom type and provider development)
provider	plugin	resource
Resource declaration	role	Role and profile module
site	Site module	Subclass
template	top scope	type

Pupput Installation

Installing Puppet Server

edure

→Steps for installation and Configuration

» Install Puppet

» Configure Server Hostname

» Start Puppet Server

→Debian, Ubuntu

Available by default

» apt-get install puppet # On clients (nodes)

» apt-get install puppetmaster # On server (master)

→RedHat, Centos, Fedora

Add EPEL repository or RHN Extra channel

» yum install puppet # On clients (nodes)

» yum install puppet-server # On server (master)

Puppet client is defalutly installed on Ubuntu machines. You can check by

```
vagrant@vagrant-ubuntu-trusty-64:~$ puppet --version
3.4.3
vagrant@vagrant-ubuntu-trusty-64:~$
```

Now we are going to uninstall it completely & Start from Stcrach

```
$ sudo apt-get purge puppet puppet-common -y
~$ sudo apt-get autoremove -y

Check the Puppet folders deleted or not
vagrant@vagrant-ubuntu-trusty-64:~$ ls /etc/puppet
ls: cannot access /etc/puppet: No such file or directory

vagrant@vagrant-ubuntu-trusty-64:~$ ls /var/lib/puppet
ls: cannot access /var/lib/puppet: No such file or directory
```

1: Enable the Puppet Package Repository

To enable the repository for Ubuntu 14.04 Trusty Tahr:

```
wget https://apt.puppetlabs.com/puppetlabs-release-trusty.deb
sudo dpkg -i puppetlabs-release-trusty.deb
sudo apt-get update
```

```
vagrant@vagrant-ubuntu-trusty-64:~/puppet$
wget https://apt.puppetlabs.com/puppetlabs-release-trusty.deb
--2018-09-28 08:46:41--  https://apt.puppetlabs.com/puppetlabs-release-trusty.deb
Connecting to 10.219.2.220:80... connected.
Proxy request sent, awaiting response... 200 OK
Length: 16944 (17K) [application/x-debian-package]
Saving to: ‘puppetlabs-release-trusty.deb’

100%[======================================================================>]
16,944      --.-K/s   in 0.003s

2018-09-28 08:46:41 (5.18 MB/s) - ‘puppetlabs-release-trusty.deb’ saved [16944/16944]
vagrant@vagrant-ubuntu-trusty-64:~/puppet$ sudo dpkg -i puppetlabs-release-trusty.deb
Selecting previously unselected package puppetlabs-release.
(Reading database ... 103605 files and directories currently installed.)
Preparing to unpack puppetlabs-release-trusty.deb ...
Unpacking puppetlabs-release (1.1-1) ...
Setting up puppetlabs-release (1.1-1) ...
vagrant@vagrant-ubuntu-trusty-64:~/puppet$ sudo apt-get update
```

2. Install Puppet Master

References

Video : https://www.youtube.com/watch?v=0yVJhb2VkVk

Chef vs Puppet : https://logz.io/blog/chef-vs-puppet/

11. Docker

Introduction

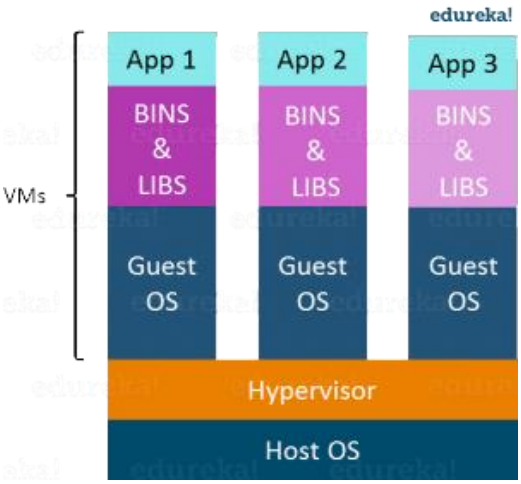
Ref : https://www.edureka.co/blog/docker-tutorial

What is Virtualization?

Virtualization is the technique of importing a Guest operating system on top of a Host operating system. This eliminated the need for extra hardware resource.

The **advantages** of Virtual Machines or Virtualization are:

- Multiple operating systems can run on the same machine
- Maintenance and Recovery were easy in case of failure conditions
- Total cost of ownership was also less due to the reduced need for infrastructure



In above pic, you can see there is a host operating system on which there are 3 guest operating systems running which is nothing but the virtual machines.

Guest OS running on top of the host OS, which will have its own kernel and set of libraries and dependencies. This takes up a large chunk of system resources, i.e. hard disk, processor and especially RAM.

Disadvantages of Virtualization:

- Running multiple Virtual Machines leads to unstable performance
- Hypervisors are not as efficient as the host operating system
- Boot up process is long and takes time

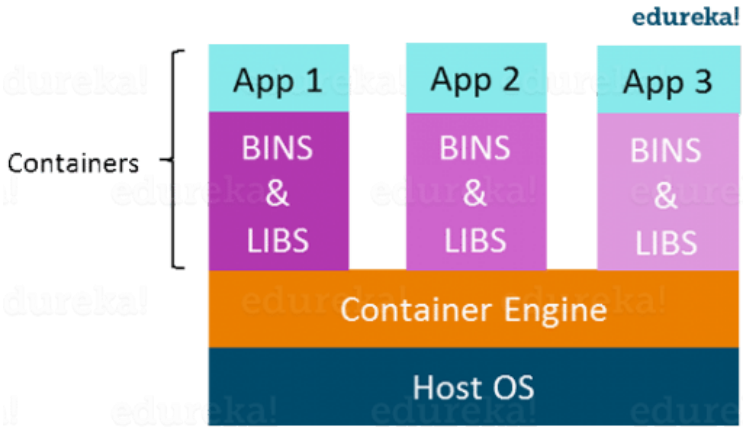
What is Containerization?

Containerization is also a type of Virtualization. Containerization is however more efficient because there is no guest OS here and utilizes a host's operating system, share relevant libraries & resources as and when needed unlike virtual machines

All the containers share, host operating system and holds only the application related binaries & libraries. They are lightweight and faster than Virtual Machines.

Advantages of Containerization over Virtualization:

- Containers on the same OS kernel are lighter and smaller
- Better resource utilization compared to VMs
- Boot-up process is short and takes few seconds

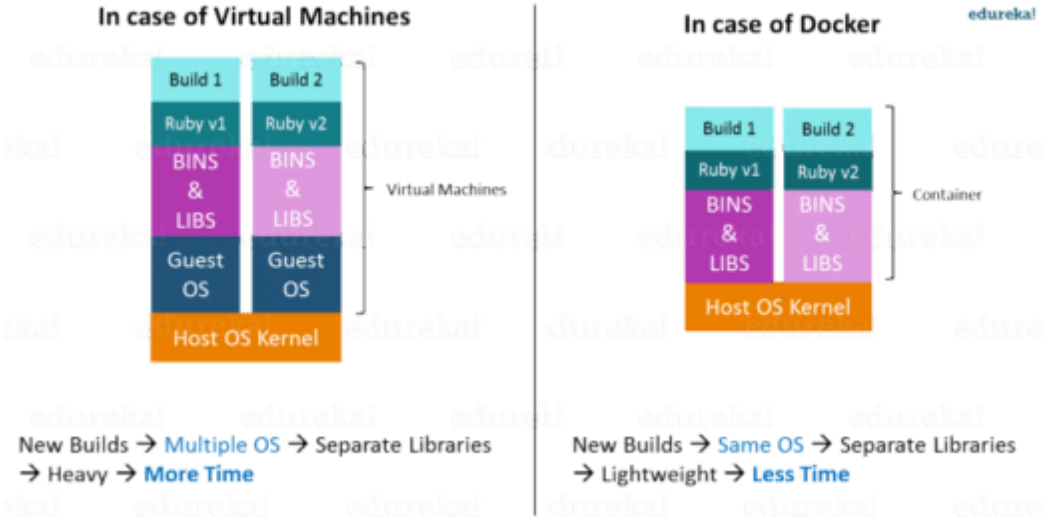


In the diagram , you can see that there is a host operating system which is shared by all the containers. Containers only contain application specific libraries which are separate for each container and they are faster and do not waste any resources.

Virtualization vs Containerization

Virtualization and Containerization both let you run multiple operating systems inside a host machine.

Virtualization deals with creating many operating systems in a single host machine. Containerization on the other hand will create multiple containers for every type of application as required



As we can see from the image, the major difference is that there are multiple Guest Operating Systems in Virtualization which are absent in Containerization. The best part of Containerization is that it is very light weight as compared to the heavy virtualization

What is Docker

Ref : https://www.edureka.co/blog/what-is-docker-container

Before we go ahead, let me summarize the learning till now:

- Virtual Machines are slow and takes a lot of time to boot.
- Containers are fast and boots quickly as it uses host operating system and shares the relevant libraries.
- Containers does not waste or block host resources unlike virtual machines.
- Containers have isolated libraries and binaries specific to the application they are running.
- Containers are handled by Containerization engine.
- Docker is one of the containerization platforms which can be used to create and run containers.

What is Docker ? – Docker is a containerization platform that packages your application and all its dependencies together in the form of a docker container to ensure that your application works seamlessly in any environment

What is Container ? – Docker Container is a standardized unit which can be created on the fly to deploy a particular application or environment. It could be an Ubuntu container, CentOs container, etc. to full-fill the requirement from an operating system point of view. Also, it could be an application oriented container like CakePHP container or a Tomcat-Ubuntu container etc.

Let's understand it with an example:

A company needs to develop a Java Application. In order to do so the **developer** will setup an environment with tomcat server installed in it. Once the application is developed, it needs to be tested by the tester.

Now the **tester** will again set up tomcat environment from the scratch to test the application. Once the application testing is done, it will be deployed on the production server.

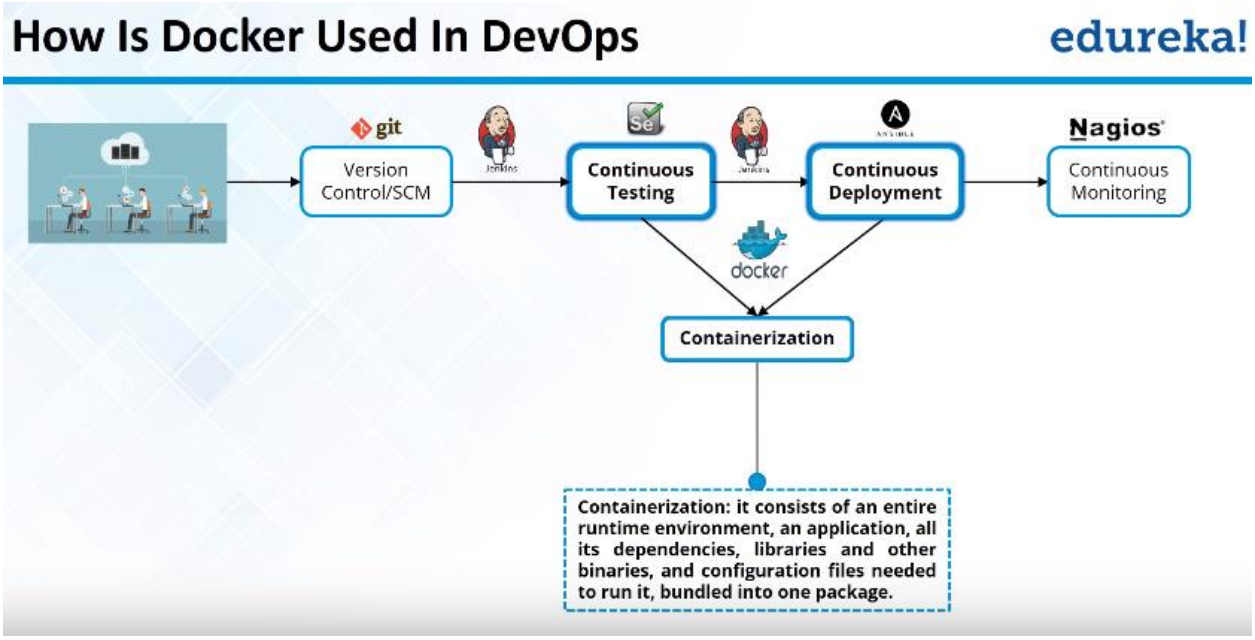
Again the **production** needs an environment with tomcat installed on it, so that it can host the Java application. **If you see the same tomcat environment setup is done thrice**. There are some issues that I have listed below with this approach:

1. There is a loss of time and effort.
2. There could be a version mismatch in different setups i.e. the developer & tester may have installed tomcat 7, however the system admin installed tomcat 9 on the production server.

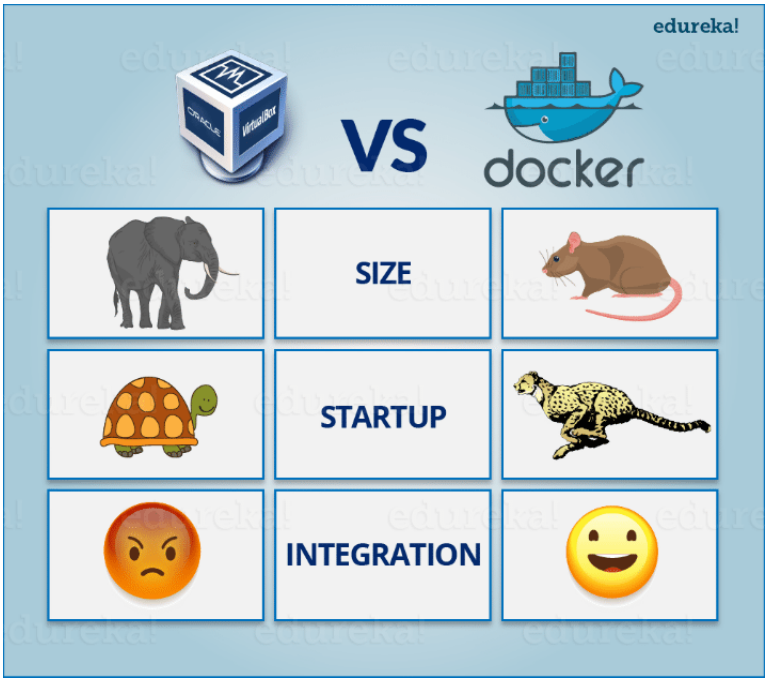
Now, I will show you how Docker container can be used to prevent this loss.

In this case, the developer will create a tomcat docker image (A Docker Image is nothing but a blueprint to deploy multiple containers of the same configurations) using a base image like Ubuntu, which is already existing in Docker Hub (Docker Hub has some base docker images available for free) .

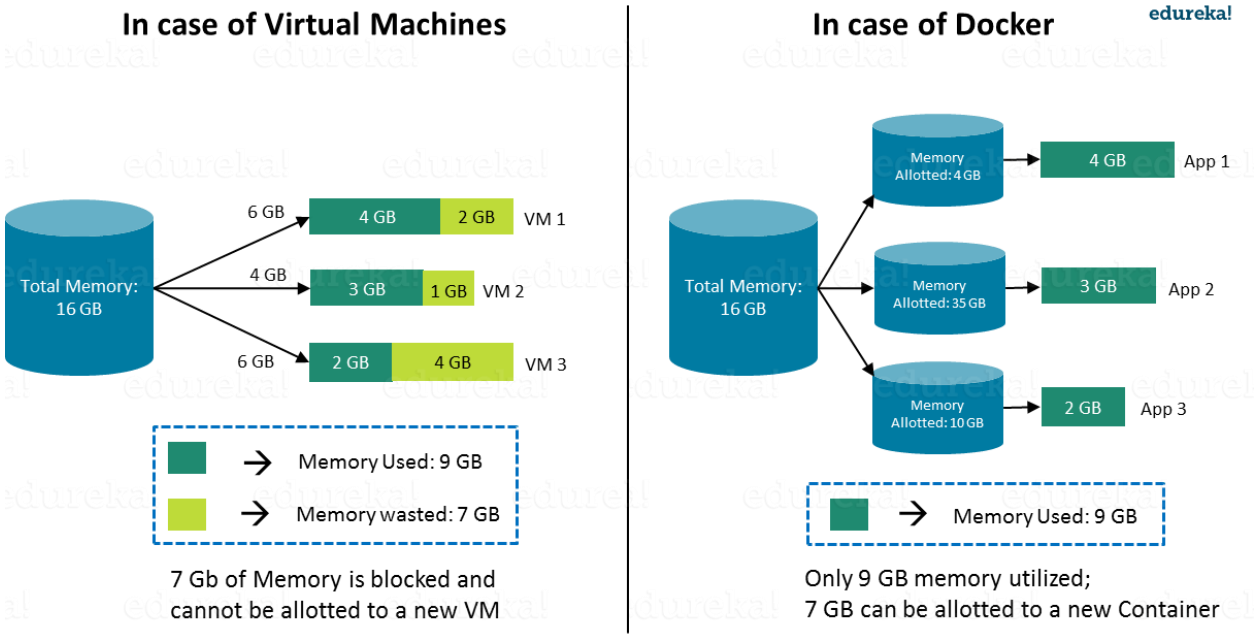
Now this image can be used by the developer, the tester and the system admin to deploy the tomcat environment. This is how docker container solves the problem.



Let's see a comparison between a Virtual machine and Docker Container to understand this better.



Size: The following image explains how Virtual Machine and Docker Container utilizes the resources allocated to them.

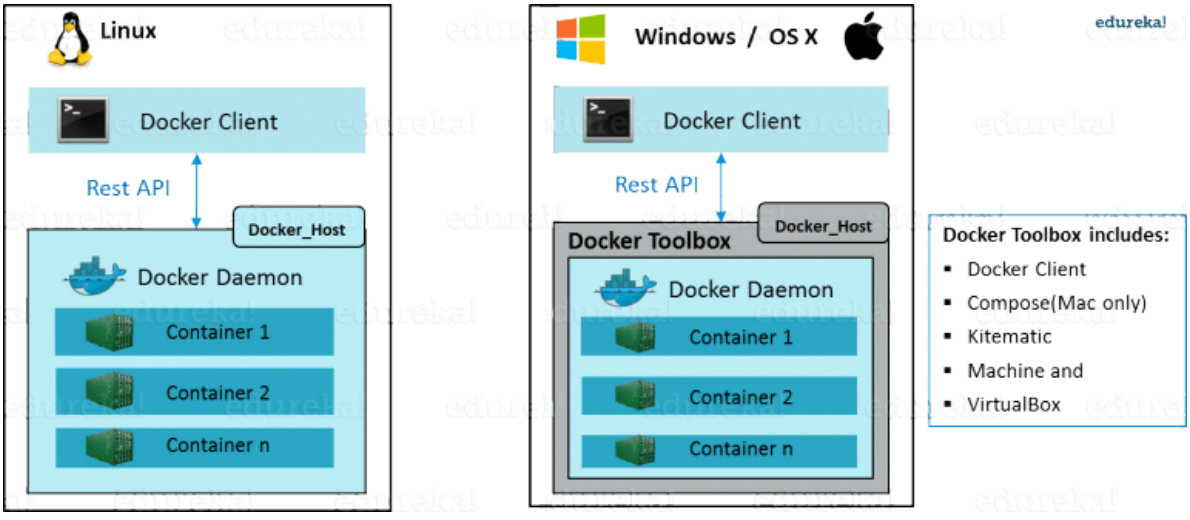


How Docker Works

Docker Engine

Docker Engine is simply the docker application that is installed on your host machine. It works like a client-server application which uses:

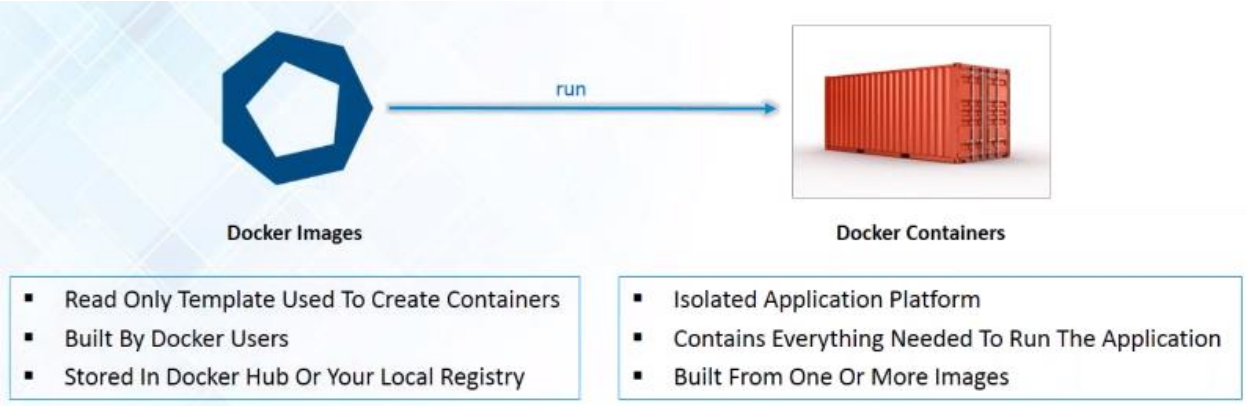
- A **server** which is a type of long-running program called a daemon process
- A command line interface (CLI) **client**
- REST API is used for communication between the CLI client and Docker Daemon



As per the above image, in a Linux Operating system, there is a Docker client which can be accessed from the terminal and a Docker Host which runs the Docker Daemon. We build our Docker images and run Docker containers by passing commands from the CLI client to the Docker Daemon.

Docker Image

Docker Image can be compared to a template which is used to create Docker Containers. They are the building blocks of a Docker Container. These Docker Images are created using the build command. These Read only templates are used for creating containers by using the run command.



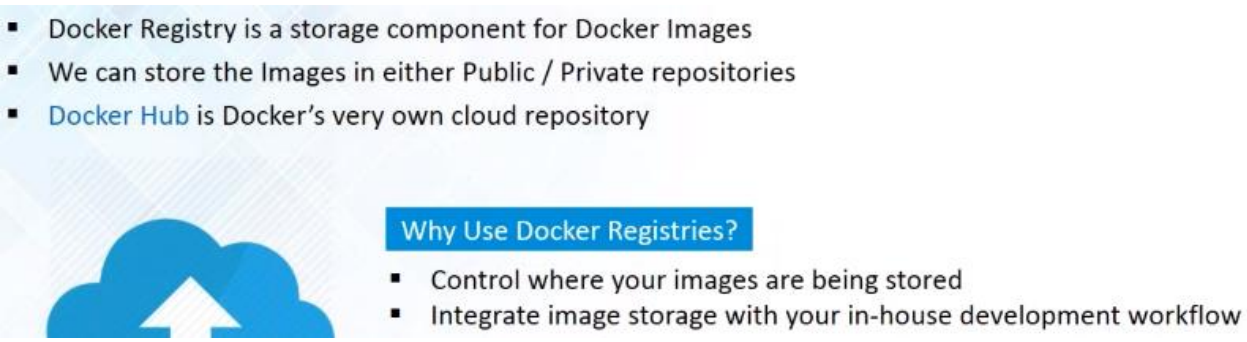
Docker Container

Containers are the ready applications created from Docker Images or you can say a Docker Container is a running instance of a Docker Image and they hold the entire package needed to run the application. This happens to be the ultimate utility of Docker.

Docker Registry?

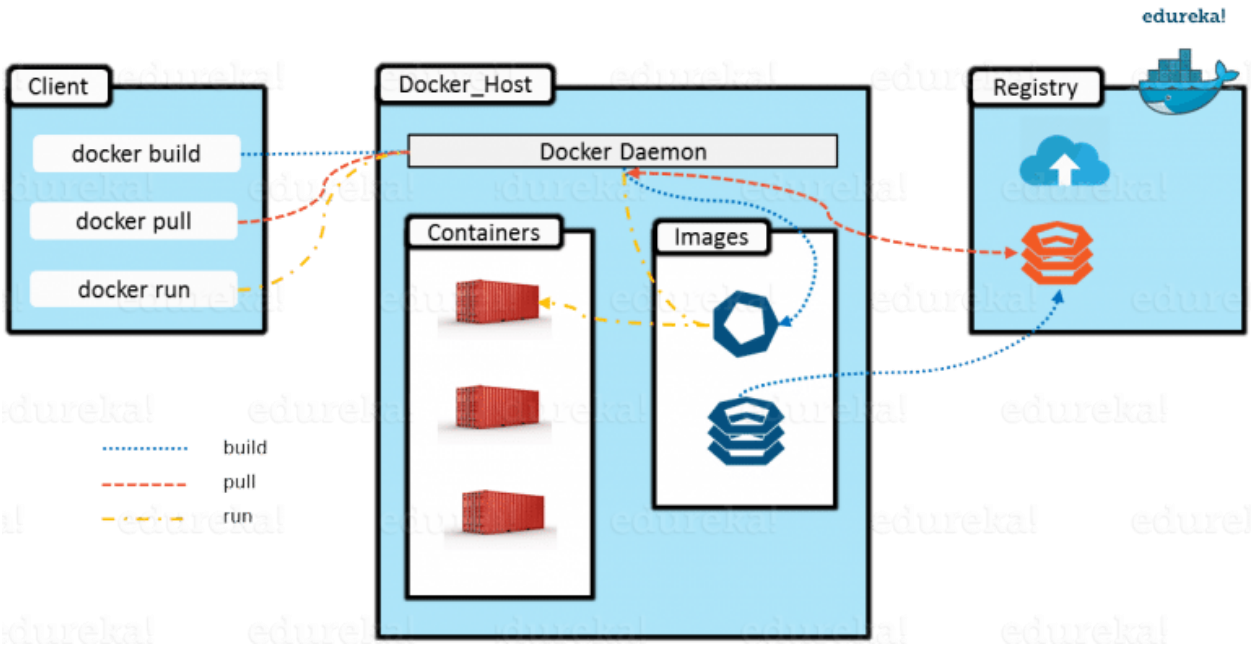
Docker Registry is where the Docker Images are stored. The Registry can be either a user's local repository or a public repository like a Docker Hub allowing multiple users to collaborate in building an application.

Even with multiple teams within the same organization can exchange or share containers by uploading them to the Docker Hub. Docker Hub is Docker's very own cloud repository similar to GitHub.



Docker Architecuture

Docker Architecture includes a Docker client – used to trigger Docker commands, a Docker Host – running the Docker Daemon and a Docker Registry – storing Docker Images. The Docker Daemon running within Docker Host is responsible for the images and containers.



- To build a Docker Image, we can use the CLI (client) to issue a build command to the Docker Daemon (running on Docker_Host). The Docker Daemon will then build an image based on our inputs and save it in the Registry, which can be either Docker hub or a local repository
- If we do not want to create an image, then we can just pull an image from the Docker hub, which would have been built by a different user
- Finally, if we have to create a running instance of my Docker image, we can issue a run command from the CLI, which will create a Docker Container.

Docker Commands

Ref: https://www.edureka.co/blog/docker-commands/

1.**docker –version**

This command is used to get the currently installed version of docker

2. docker pull

Usage: docker pull <image name>

This command is used to pull images from the **docker repository**(hub.docker.com)

3. **docker run**

Usage: docker run -it -d <image name>

This command is used to create a container from an image

4. **docker ps**

This command is used to list the running containers

5. **docker ps -a**

This command is used to show all the running and exited containers

6. **docker exec**

Usage: docker exec -it <container id> bash

This command is used to access the running container

7. **docker stop**

Usage: docker stop <container id>

This command stops a running container

8. **docker kill**
Usage: docker kill <container id>

This command kills the container by stopping its execution immediately. The difference between 'docker kill' and 'docker stop' is that 'docker stop' gives the container time to shutdown gracefully, in situations when it is taking too much time for getting the container to stop, one can opt to kill it

9. **docker commit**

Usage: docker commit <conatainer id> <username/imagename>

This command creates a new image of an edited container on the local system

10. **docker login**

This command is used to login to the docker hub repository

11. **docker push**

Usage: docker push <username/image name>

This command is used to push an image to the docker hub repository

12. **docker images**

This command lists all the locally stored docker images

13. **docker rm**

Usage: docker rm <container id>

This command is used to delete a stopped container

14. **docker rmi**

Usage: docker rmi <image-id>

This command is used to delete an image from local storage

15. **docker build**

Usage: docker build <path to docker file>

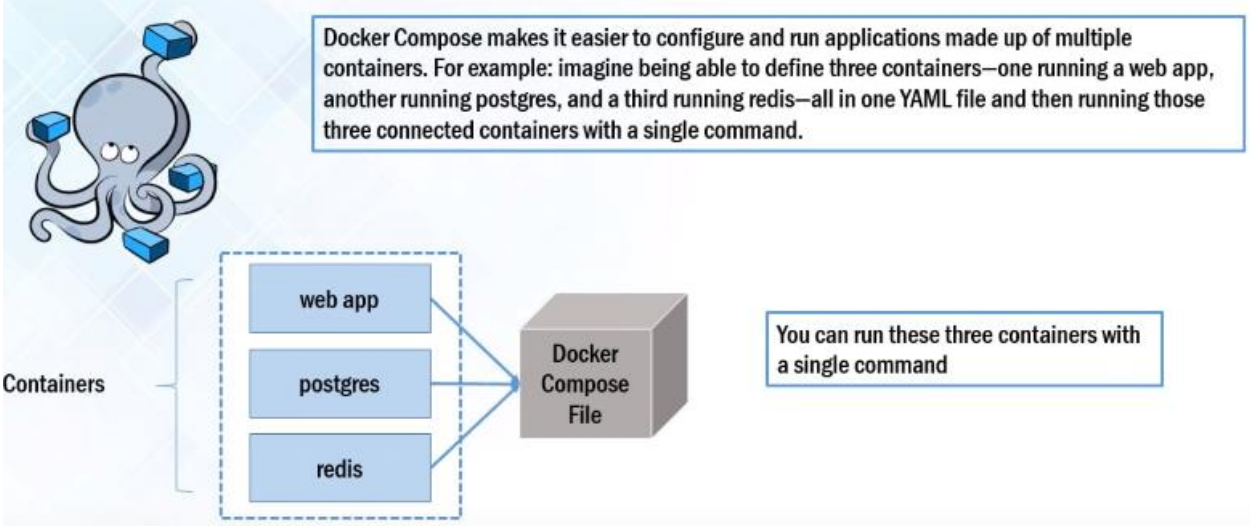
This command is used to build an image from a specified docker file

Installing Docker on Ubuntu

https://www.youtube.com/watch?v=lcOfORDAMpQ&list=PL9ooVrP1hQOHUKuqGuiWLQoJ-LD25KxI5&index=2 19: min

Docker Compose

when I had to containerize multiple services in separate containers, who can you communicate beween them ? & How can start them with Single Operations. That Stage Docker Compose Come into the picture.



Docker Compose can be used to create separate containers and host them for each of the stacks in a Full stack application which contains MongoDB Express Angular & NodeJs.

By using Docker Compose, we can host each of these technologies in separate containers on the same host and get them to communicate with each other. Each container will expose a port for communicating with other containers.

The communication and up-time of these containers will be maintained by Docker Compose

HandsOn

https://www.edureka.co/blog/install-docker/

Install : https://www.youtube.com/watch?v=lcQfQRDAMpQ&list=PL9ooVrP1hQOHUKuqGuiWLQoJ-LD25KxI5&index=2

Edureka : https://www.youtube.com/watch?v=h0NCZbHjIpY&list=PL9ooVrP1hQOHUKuqGuiWLQoJ-LD25KxI5

Next : https://www.youtube.com/watch?v=wi-MGFhrad0&list=PLhW3qG5bs-L99pQsZ74f-LC-tOEsBp2rK

Docker Install Ubuntu

1.First, update your existing list of packages:

```
sudo apt update
```

2.Next, install a few prerequisite packages which let apt use packages over HTTPS:

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

3.Then add the GPG key for the official Docker repository to your system:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4.Add the Docker repository to APT sources:

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
bionic stable"
```

5.Next, update the package database with the Docker packages from the newly added repo:

```
sudo apt update
```

6.Make sure you are about to install from the Docker repo instead of the default Ubuntu repo:

```
apt-cache policy docker-ce
```

You'll see output like this, although the version number for Docker may be different:

```
docker-ce:
  Installed: (none)
  Candidate: 18.03.1~ce~3-0~ubuntu
  Version table:
     18.03.1~ce~3-0~ubuntu 500
```

```
        500 https://download.docker.com/linux/ubuntu bionic/stable amd64 Packages
```

7.Notice that docker-ce is not installed, but the candidate for installation is from the Docker repository for Ubuntu 18.04 (bionic).So, install Docker:

```
sudo apt install docker-ce
```

8.Docker should now be installed, the daemon started, and the process enabled to start on boot. Check that it's running:

```
sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2018-10-01 21:10:48 IST; 3min 39s ago
```

if not start, start the service by running

```
sudo service docker start
```

```
satya@satya:~/.../docker$ docker --version
Docker version 18.06.1-ce, build e68fc7a
```

Task 1 : pull centos image from hub.docker

Pull image from Docker Hub

```
satya@satya:~/.../docker$ sudo docker pull centos
Using default tag: latest
latest: Pulling from library/centos
256b176beaff: Pull complete
Digest: sha256:6f6d986d425aeabdc3a02cb61c02abb2e78e57357e92417d6d58332856024faf
Status: Downloaded newer image for centos:latest
```

First, it will check the local registry for CentOS image. If it doesn't find there, then it will go to the docker hub and pull the image

Check Downloaded Image

-By Def. location is /var/lib/docker

-In the case of aufs:/var/lib/docker/aufs/diff/<id>

-In the case of devicemapper:/var/lib/docker/devicemapper/devicemapper/data

Now, Run the CentOS container.

```
satya@satya:~/.../docker$ sudo docker run -it centos
[root@15458942452c /]#
You logged in centos successfully
```

==========

Task 2 : Wordpress + MySQL + PhpMyAdmin

==========

Basically, you need one container for WordPress and you need one more container as MySQL for back end, that MySQL container should be linked to the wordpress container. We also need one more container for Php Myadmin that will be linked to MySQL database, basically, it is used to access MySQL database.

[Worpress]

|

link= wordprees+ MySQL

|

[MySQL]

|

link= MySql+ PhpMyAdmin

|

[PhpMyAdmin]

Here we will write Docker Compose file to install & make link between them

Steps involved:

1. Install Docker Compose:
2. Install WordPress: We'll be using the official WordPress and MariaDB Docker images.
3. Install MariaDB: MariaDB is a database it provides an SQL interface for accessing data.
4. Install PhpMyAdmin: handle the administration of MySQL over the Web.
5. Create The WordPress Site

1. Install Docker Compose

Install Python Pip first:

sudo apt-get install python-pip -y

Now, you can install Docker Compose:

sudo pip install docker-compose

2. Install WordPress:

Create a wordpress directory:

```
mkdir wordpress
cd wordpress/
```

In this directory create a Docker Compose YAML file, then edit it using gedit:

```
sudo gedit docker-compose.yml
-------------------------------------------------------
version: "2"
services:
  my-wpdb:
    image: mariadb
    ports:
      - "8081:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
  my-wp:
    image: wordpress
    volumes:
      - ./:/var/www/html
    ports:
      - "8080:80"
    links:
      - my-wpdb:mysql
    environment:
      WORDPRESS_DB_PASSWORD: root
  phpmyadmin:
    image: corbinu/docker-phpmyadmin
    links:
      - my-wpdb:mysql
    ports:
      - 8181:80
    environment:
      MYSQL_USERNAME: root
      MYSQL_ROOT_PASSWORD: root
-------------------------------------------------------
```

Now start the application group:

```
satya@satya:~/.../Wordpress$ docker-compose up -d
```

ERROR: Couldn't connect to Docker daemon at http+docker://localhost - is it running?

this error means you don't have enogh permissions. so run with Sudi

satya@satya:~/.../Wordpress$ sudo docker-compose up -d

Test wordpress

http://localhost:8080/

Test PhpMyAdmin

http://localhost:8181/

==========

Docker AWS Installation

==========

https://www.youtube.com/watch?v=HqBMEmoAd1M&index=7&list=PLhW3qG5bs-L99pOsZ74f-LC-tOEsBp2rK

Using Amazon Linux AMI

---

sudo yum -y update

sudo yum install -y docker

sudo service docker status

sudo service docker start

sudo service docker stop

sudo docker info

sudo usermod -a -G docker ec2-user

sudo docker images

sudo docker ps -a

docker run <image_name>

sudo docker run hello-world

sudo yum remove docker

getdocker.com

==========

Docker COmmands

==========

basic

---

sudo docker version

sudo docker -v

sudo docker info

sudo docker --help

sudo docker login

Images

---

sudo docker images

sudo docker pull

sudo docker rmi

sudo docker

sudo docker

Conatiners

----------=

sudo docker ps

sudo docker run

sudo docker start

sudo docker stop

System

----

sudo docker stats

sudo docker system df

sudo docker system prune

Create Docker Images

[root@ip-172-31-22-216 ec2-user]# docker pull ubuntu

Using default tag: latest

latest: Pulling from library/ubuntu

124c757242f8: Pull complete

9d866f8bde2a: Pull complete

fa3f2f277e67: Pull complete

398d32b153e8: Pull complete

afde35469481: Pull complete

Digest: sha256:de774a3145f7ca4f0bd144c7d4ffb2931e06634f11529653b23eba85aef8e378

Status: Downloaded newer image for ubuntu:latest

[root@ip-172-31-22-216 ec2-user]# docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world	latest	4ab4c602aa5e	3 weeks ago	1.84kB

ubuntu latest cd6d8154f1e1 3 weeks ago 84.1MB

To get Specfic version

[root@ip-172-31-22-216 ec2-user]# docker pull ubuntu:18.04

18.04: Pulling from library/ubuntu

Digest: sha256:de774a3145f7ca4f0bd144c7d4ffb2931e06634f11529653b23eba85aef8e378

Status: Downloaded newer image for ubuntu:18.04

By running Docker image, we are creating Container

docker run --name MyDockerUbuntu -it ubuntu bash

[root@ip-172-31-22-216 ec2-user]# sudo docker run -it ubuntu

root@76baecadb14d:/#

root@76baecadb14d:/# ls

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

root@76baecadb14d:/#

Create Docker Containers

Dockerfile :

A text file with instructions to build image

Automation of Docker Image Creation

FROM

RUN

CMD

Step 1 : Create a file named Dockerfile

Step 2 : Add instructions in Dockerfile

Step 3 : Build dockerfile to create image

Step 4 : Run image to create container

Containers are the Running instances of Images.

When ever we run the image, it creates a Container

Create Our Own Imgage

---

create a Directory /Docker, write a Dockerfile under it

Dockerfile

#Getting base image Ubuntu

FROM ubuntu

MAINTAINER smlcodes<smlcodes@gmail.com>

RUN apt-get update

CMD ["echo" , "Hello, Im Satya"]

Create Docker Image

Go to /Docker location & Run

docker build -t smlcodes1:1.0 .

[root@ip-172-31-22-216 Docker]# docker build -t smlcodes1:1.0 .

Sending build context to Docker daemon 2.048kB

Step 1/4 : FROM ubuntu

 ---> cd6d8154f1e1

Step 2/4 : MAINTAINER smlcodes<smlcodes@gmail.com>

 ---> Running in 53f4f543c194

Removing intermediate container 53f4f543c194

 ---> 78dd08c39f7b

Step 3/4 : RUN apt-get update

 ---> Running in 4d6c57bf6859

Successfully built 863bc0cd3f7c

Successfully tagged smlcodes1:1.0

Check Image in the List

If you check the list image is created locally.

image just contains a program to print hello

[root@ip-172-31-22-216 Docker]# sudo docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
smlcodes1	1.0	863bc0cd3f7c	4 minutes ago	127MB
hello-world	latest	4ab4c602aa5e	3 weeks ago	1.84kB
ubuntu	18.04	cd6d8154f1e1	3 weeks ago	84.1MB
ubuntu	latest	cd6d8154f1e1	3 weeks ago	84.1MB

Run your image, it will create container

[root@ip-172-31-22-216 Docker]# sudo docker run -it smlcodes1

Unable to find image 'smlcodes1:latest' locally

docker: Error response from daemon: pull access denied for smlcodes1, repository does not exist or may require 'docker login'.

See 'docker run --help'

If You got this error Login with Your Docker account or use id to Run [sudo docker run -it 863bc0cd3f7c]

[root@ip-172-31-22-216 Docker]# docker login

Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.

Username: smlcodes

Password:

[root@ip-172-31-22-216 Docker]# sudo docker run -it smlcodes1:1.0

Hello, Im Satya

Docker Compose

Docker compose

: tool for defining & running multi-container docker applications

: use yaml files to configure application services (docker-compose.yml)

: can start all services with a single command : docker compose up

: can stop all services with a single command : docker compose down

: can scale up selected services when required

Step 1 : install docker compose

(already installed on windows and mac with docker)

docker-compose -v

2 Ways

1. https://github.com/docker/compose/rel...

2. Using PIP

pip install -U docker-compose

Step 2 : Create docker compose file at any location on your system

```
   docker-compose.yml
------------------------
[root@ip-172-31-22-216 Docker]# cat docker-compose.yml
version: '3.3'
```

```
services:

  web:
    image: nginx

  database:
    image: redis
```

Step 3 : Check the validity of file by command

docker-compose config

Step 4 : Run docker-compose.yml file by command

sudo docker-compose up -d

Steps 5 : Bring down application by command

docker-compose down

TIPS

How to scale services

—scale

docker-compose up -d --scale database=4

Docker Volumes

----------=-------

-By default all files created inside a container are stored on a writable container layer

-The data doesn't persist when that container is no longer running

-A container's writable layer is tightly coupled to the host machine where the container is running. You can't easily move the data somewhere else.

-Docker has two options for containers to store files in the host machine so that the files are persisted even after the container stops

Use of Volumes

==========

Decoupling container from storage

Share volume (storage/data) among different containers

Attach volume to container

On deleting container volume does not delete

: docker volume //get information

: docker volume create

: docker volume ls

: docker volume inspect

: docker volume rm

: docker volume prune

VOLUMES and BIND MOUNTS

- -Volumes are stored in a part of the host filesystem which is managed by Docker
- -Non-Docker processes should not modify this part of the filesystem
- -Bind mounts may be stored anywhere on the host system
- -Non-Docker processes on the Docker host or a Docker container can modify them at any time
- -In Bind Mounts, the file or directory is referenced by its full path on the host machine.
- -Volumes are the best way to persist data in Docker
- -volumes are managed by Docker and are isolated from the core functionality of the host machine
- -A given volume can be mounted into multiple containers simultaneously.

- -When no running container is using a volume, the volume is still available to Docker and is not removed automatically. You can remove unused volumes using docker volume prune.
- -When you mount a volume, it may be named or anonymous.
- -Anonymous volumes are not given an explicit name when they are first mounted into a container
- -Volumes also support the use of volume drivers, which allow you to store your data on remote hosts or cloud providers, among other possibilities.

Create Volume

satya@satya:~/.../docker$ sudo docker volume create myvol1

myvol1

list the docker volumes

satya@satya:~/.../docker$ sudo docker volume ls

DRIVER VOLUME NAME

local 046cd7015ac74c659beb1f8d93293993161f1e4103715cdc29c4b447a105dcf2

local 3c6e5051ff9c98b23829c45e4a374f1426f7bbd15eab39d52720f1d8337dea82

local 9c8f01a4316e809eee624833b4538afc39e7e8194e81ee792e8ce5873b87298e

local myvol1

satya@satya:~/.../docker$ sudo docker volume inspect myvol1


```
[
    {
        "CreatedAt": "2018-10-02T19:42:26+05:30",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/myvol1/_data",
        "Name": "myvol1",
        "Options": {},
        "Scope": "local"
    }
]
```

to remove volume

satya@satya:~/.../docker$ sudo docker volume rm -f myvol1

to remove all unused volumes

sudo docker volume prune

==========

Example - using Jenkins

==========

1.pull Jenkins

docker pull jenkins

2.Now i want to store the all /var/.jenkins related data to 'myvol1', so that if we delete

jenkins container , the jenkins data will be available, to do so run below cmd

sudo docker run -p 8080:8080 -p 50000:50000 jenkins

This command will directly store the data into the container only

satya@satya:~/.../docker$ sudo docker run --name MyJenkins1 -v myvol1:/var/jenkins_home -p 8080:8080 -p 50000:50000 jenkins

Here all jenkins data of 'MyJenkins1' will be stored at myvol1

IF We start the another Jenkins instance with same volume location, docker wont create another

jenkins intsnace, instead ot will share the same data to newly added jenkins with another port

satya@satya:~/.../docker$ sudo docker run --name MyJenkins2 -v myvol1:/var/jenkins_home -p 9090:8080 -p 50000:50000 jenkins

Commands

docker run --name MyJenkins1 -v myvol1:/var/jenkins_home -p 8080:8080 -p 50000:50000 jenkins

docker run --name MyJenkins2 -v myvol1:/var/jenkins_home -p 9090:8080 -p 60000:50000 jenkins

Bind Physical Location

---

docker run --name MyJenkins3 -v /Users/raghav/Desktop/Jenkins_Home:/var/jenkins_home -p 9191:8080 -p 40000:50000 jenkins

References

https://hub.docker.com/_/jenkins/

https://docs.docker.com/storage/volumes/

==========

Docker Swarm

==========

-A swarm is a group of machines that are running Docker and joined into a cluster.

-Docker Swarm is a tool for Container Orchestration

Let's take an example

You have 100 containers,You need to do

- Health check on every container

- Ensure all containers are up on every system

- Scaling the containers up or down depending on the load

- Adding updates/changes to all the containers

Orchestration - managing and controlling multiple docker containers as a single service

Tools available - Docker Swarm, Kubernetes, Apache Mesos

Pre-requisites

1. Docker 1.13 or higher

2. Docker Machine (pre installed for Docker for Windows and Docker for Mac)https://docs.docker.com/machine/insta...

https://docs.docker.com/get-started/p...

Step 1 : Create Docker machines (to act as nodes for Docker Swarm) Create one machine as manager and others as workers

docker-machine create --driver hyperv manager1 docker-machine create --driver virtualbox manager1

docker-machine:Error with pre-create check: "exit status 126"

https://stackoverflow.com/questions/3...

brew cask install virtualbox;

Create one manager machine

and other worker machines

Step 2 : Check machine created successfully

docker-machine ls

docker-machine ip manager1

Step 3 : SSH (connect) to docker machine

docker-machine ssh manager1

Step 4 : Initialize Docker Swarm docker swarm init --advertise-addr MANAGER_IP

docker node ls

(this command will work only in swarm manager and not in worker)

Step 5 : Join workers in the swarm

Get command for joining as worker

In manager node run command

docker swarm join-token worker

This will give command to join swarm as worker

docker swarm join-token manager

This will give command to join swarm as manager

SSH into worker node (machine) and run command to join swarm as worker

In Manager Run command - docker node ls to verify worker is registered and is ready

Do this for all worker machines

Step 6 : On manager run standard docker commands

docker info

check the swarm section

no of manager, nodes etc

Now check docker swarm command options

docker swarm

Step 7 : Run containers on Docker Swarm

docker service create --replicas 3 -p 80:80 --name serviceName nginx

Check the status:

docker service ls

docker service ps serviceName

Check the service running on all nodes

Check on the browser by giving ip for all nodes

Step 8 : Scale service up and down

On manager node

docker service scale serviceName=2

Inspecting Nodes (this command can run only on manager node)

docker node inspect nodename

docker node inspect self

docker node inspect worker1

Step 9 : Shutdown node

docker node update --availability drain worker1

Step 10 : Update service

docker service update --image imagename:version web

docker service update --image nginx:1.14.0 serviceName

Step 11 : Remove service

docker service rm serviceName

docker swarm leave : to leave the swarm

docker-machine stop machineName : to stop the machine

docker-machine rm machineName : to remove the machine

REFERENCES:

https://docs.docker.com/get-started/p...

https://rominirani.com/docker-swarm-t...

FAQs & Helpful Tips:

- A swarm is a group of machines that are running Docker and joined into a cluster
- A cluster is managed by swarm manager
- The machines in a swarm can be physical or virtual. After joining a swarm, they are referred to as nodes
- Swarm managers are the only machines in a swarm that can execute your commands, or authorise other machines to join the swarm as workers
- Workers are just there to provide capacity and do not have the authority to tell any other machine what it can and cannot do
- you can have a node join as a worker or as a manager. At any point in time, there is only one LEADER and the other manager nodes will be as backup in case the current LEADER opts out

References

https://www.youtube.com/watch?v=h0NCZbHjIpY&list=PL9ooVrP1hQOHUKuqGuiWLQoJ-LD25KxI5

Docker Tutorial - Introduction To Docker & Containerization

What is Docker Container? - Containerize Your Application Using Docker

What Is Docker & Docker Container ? A Deep Dive Into Docker !

Top 15 Docker Commands - Docker Commands Tutorial

Install Docker - Docker Installation On Ubuntu And CentOS

Docker Networking - Explore How Containers Communicate With Each Other

Docker Swarm For Achieving High Availability

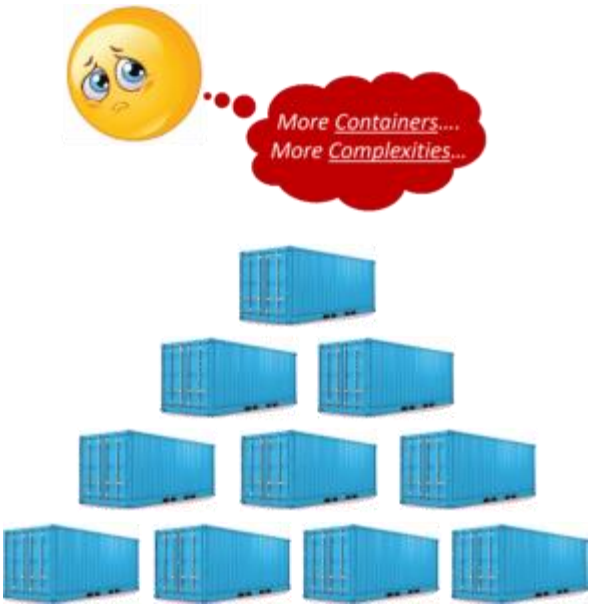
Docker Compose For Containerizing A MEAN Stack Application

Kubernetes vs Docker: Comparing The Two Container Orchestration Giants!

12. Kubernates

Kubernetes is an open-source **container management (orchestration) tool**. It's container management responsibilities include container deployment, scaling & descaling of containers & container load balancing.

It was originally **designed by Google** and is now maintained by the Cloud Native Computing Foundation



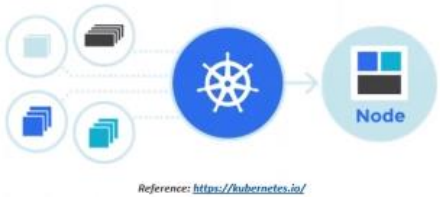
1. Containers could not **communicate** with each other
2. Containers had to be **deployed appropriately**
3. Containers had to be **managed carefully**
4. **Auto scaling** was not possible
5. **Distributing traffic** was still challenging

Kubernetes is an open-source **Container Management** tool which automates *container deployment, container (de)scaling & container load balancing.*

Benefit**: Works brilliantly with all cloud vendors: **Public, Hybrid & On-Premises.

More About Kubernetes

- Written on Golang, it has a huge community because it was first developed by Google & later donated to **CNCF**
- Can group 'n' no of containers into one logical unit for managing & deploying them easily

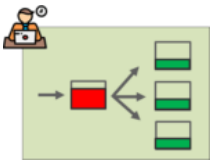


Reference: https://kubernetes.io/

Features Of Kubernetes

1. Automatic Binpacking

Kubernetes automatically packages your application and schedules the containers based on their requirements and available resources while not sacrificing availability. To ensure complete utilization and save unused resources, Kubernetes balances between critical and best-effort workloads.



2. Service Discovery & Load balancing

There is no need to worry about networking and communication because Kubernetes will automatically assign IP addresses to containers and a single DNS name for a set of containers, that can load-balance traffic inside the cluster.

3. Storage Orchestration

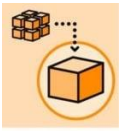
With Kubernetes, you can mount the storage system of your choice. You can either opt for local storage, or choose a public cloud provider such as GCP or AWS, or perhaps use a shared network storage system such as NFS, iSCSI, etc.

4. Self-Healing

Personally, this is my favorite feature. Kubernetes can automatically restart containers that fail during execution and kills those containers that don't respond to user-defined health checks. But if nodes itself die, then it replaces and reschedules those failed containers on other available nodes.

5. Secret & Configuration Management

Kubernetes can help you deploy and update secrets and application configuration without rebuilding your image and without exposing secrets in your stack configuration.



6. Batch Execution

In addition to managing services, Kubernetes can also manage your batch and CI workloads, thus replacing containers that fail, if desired.

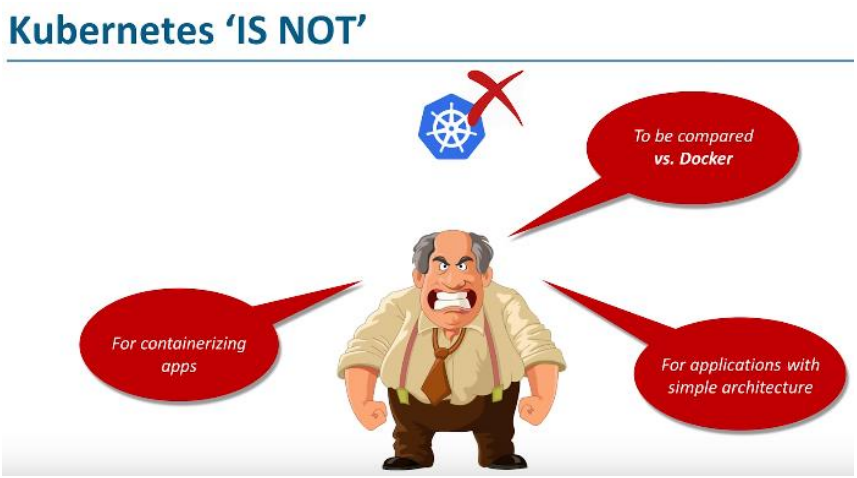
7. Horizontal Scaling

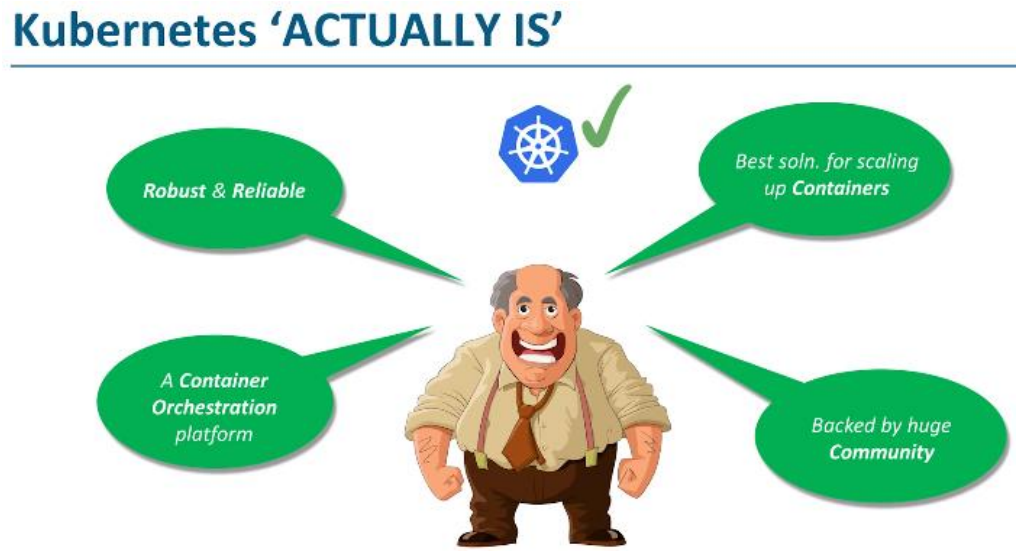
Kubernetes needs only 1 command to scale up the containers, or to scale them down when using the CLI. Else, scaling can also be done via the Dashboard (kubernetes UI).

8. Automatic Rollbacks & Rollouts

Kubernetes progressively rolls out changes and updates to your application or its configuration, by ensuring that not all instances are worked at the same instance. Even if something goes wrong, Kubernetes will rollback the change for you.

Kubernetes VS Docker





FEATURES	Kubernetes	Docker Swarm
Installation & Cluster configuration	Complicated & time consuming	Easy & fast
GUI	GUI available	GUI not available
Scalability	Scaling up is slow compared to Swarm; but guarantees stronger cluster state	Scaling up is faster than K8S; but cluster strength not as robust
Load Balancing	Load balancing requires manual service configuration	Provides built in load balancing technique
Updates & Rollbacks	Process scheduling to maintain services while updating	Progressive updates and service health monitoring throughout the update
Data Volumes	Only shared with containers in same Pod	Can be shared with any other container
Logging & Monitoring	Inbuilt logging & monitoring tools	Only 3rd party logging & monitoring tools

Kubernetes @ Pokemon GO

Pokemon Go is an augmented reality game developed by **Niantic** for Android & iOS devices.

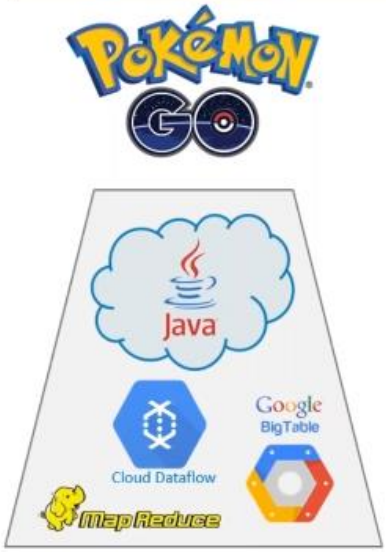
"We believe that people are healthier when they go outside and have a reason to be connected to others."

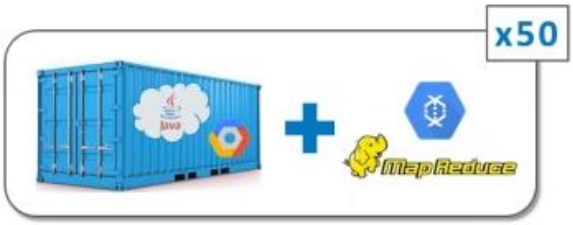
- **Edward Wu,** Director of Software Engineering, ***Niantic Labs***

★ **KEY STATS:-**

- **500+ million** downloads, **20+ million** daily active users
- Initially launched only in NA, Australia & New Zealand
- Inspired users to walk over 5.4 billion miles in a year
- Surpassed engineering expectations by 50 times

Easy Scaling Of Containers Using Kubernetes





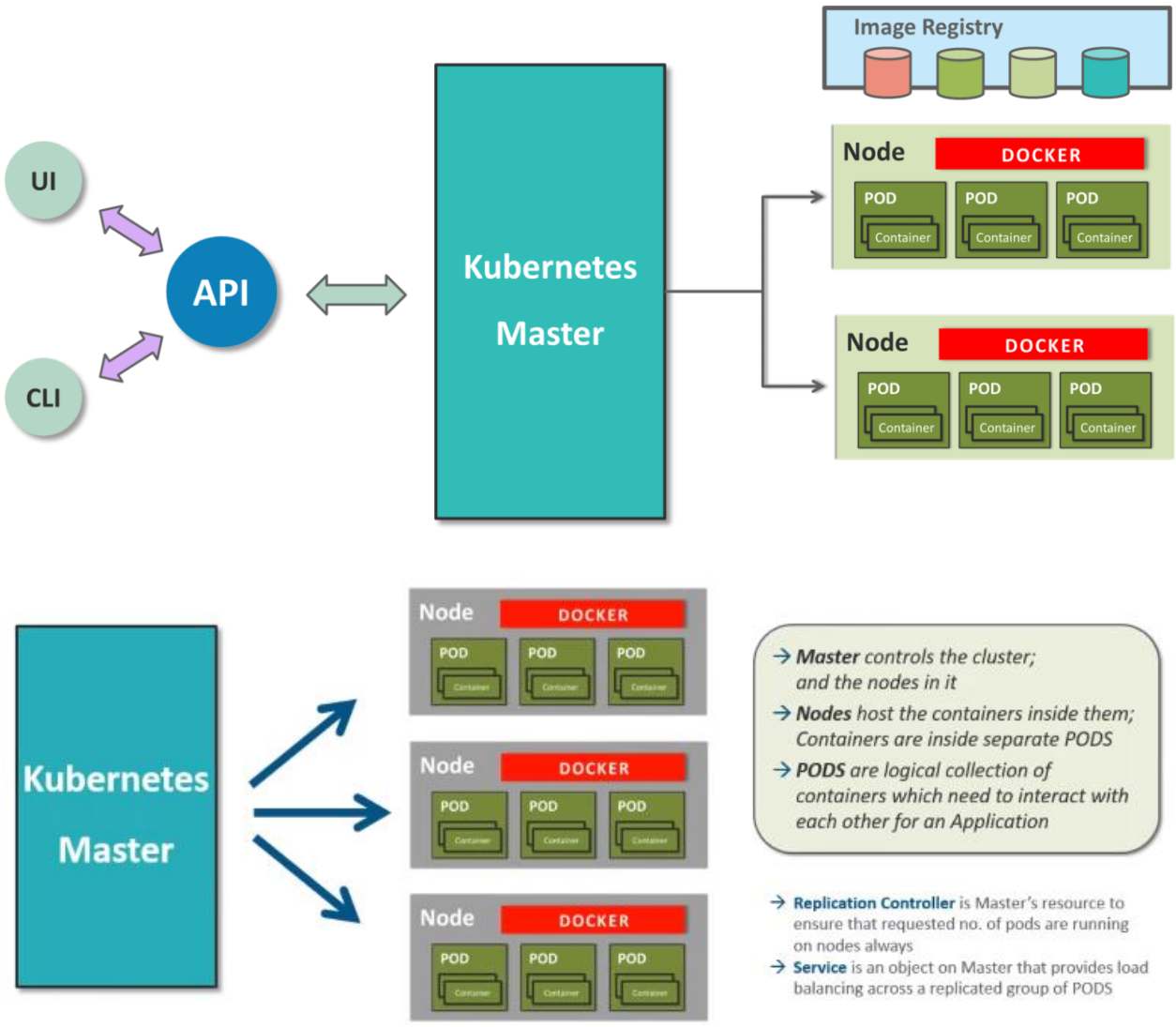
CHALLENGE

- *Biggest challenge for most applications is **horizontal scaling***
- *But for Pokemon Go, **vertical scaling** was also a major challenge, because of **real-time activity in gaming environment** from millions of users world-wide*
- *Niantic were prepared for traffic disasters of upto x5 times*

SOLUTION

- *Thanks to **Kubernetes**, Niantic were able to handle x50 times traffic*

Kubernetes Architecture



Master controls the cluster, and the nodes in it. It ensures the execution only happens in nodes and coordinates the act. Nodes host the containers; in-fact these Containers are grouped logically to form Pods. Each node can run multiple such Pods, which are a group of containers, that interact with each other, for a deployment.

Replication Controller is Master's resource to ensure that the requested no. of pods are always running on nodes. Serviceis an object on Master that provides load balancing across a replicated group of Pods.

So, that's the Kubernetes architecture in simple fashion. You can expect more details on the architecture in my next blog. A better news is, the next blog will also have a hands-on demonstration of installing Kubernetes cluster and deploying an application.

Install Kubernetes Cluster on Ubuntu

https://www.edureka.co/blog/install-kubernetes-on-ubuntu

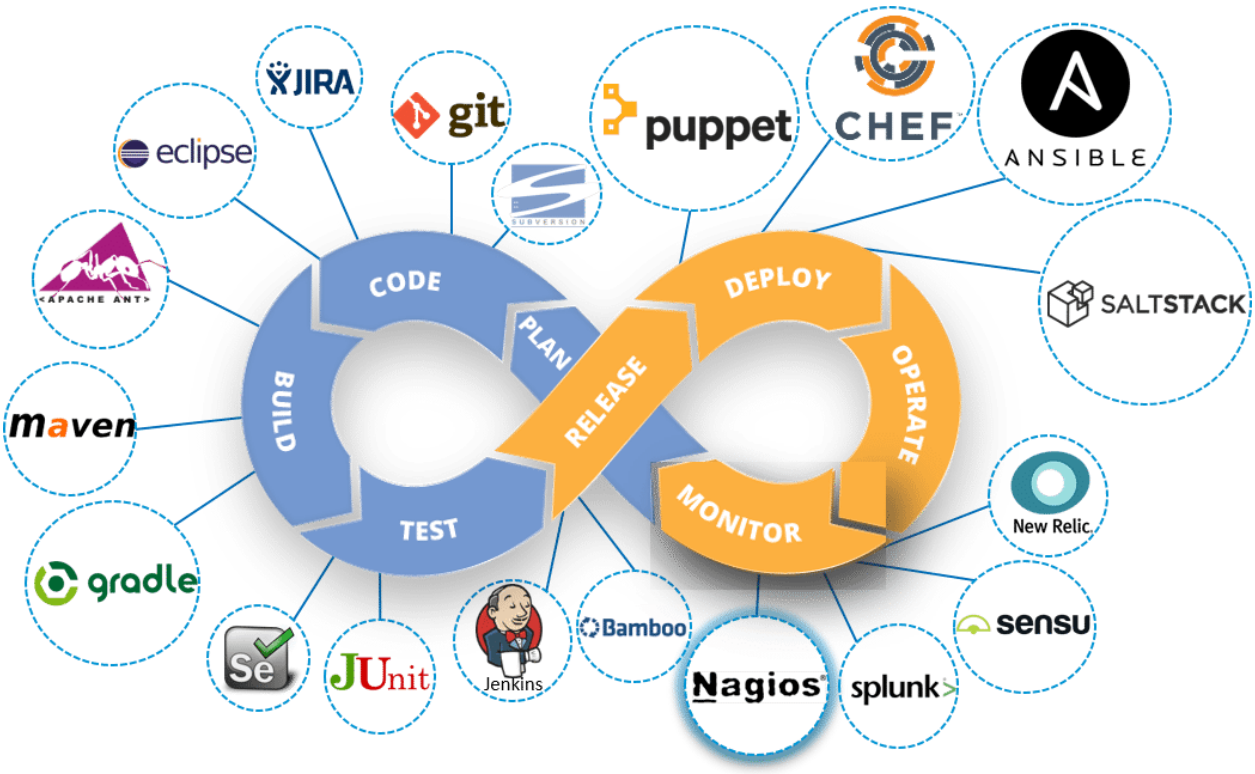
Refetenfces

Intro : https://www.edureka.co/blog/what-is-kubernetes-container-orchestration\

13.Nagios

https://www.youtube.com/watch?v=qehuAgKHFQ0

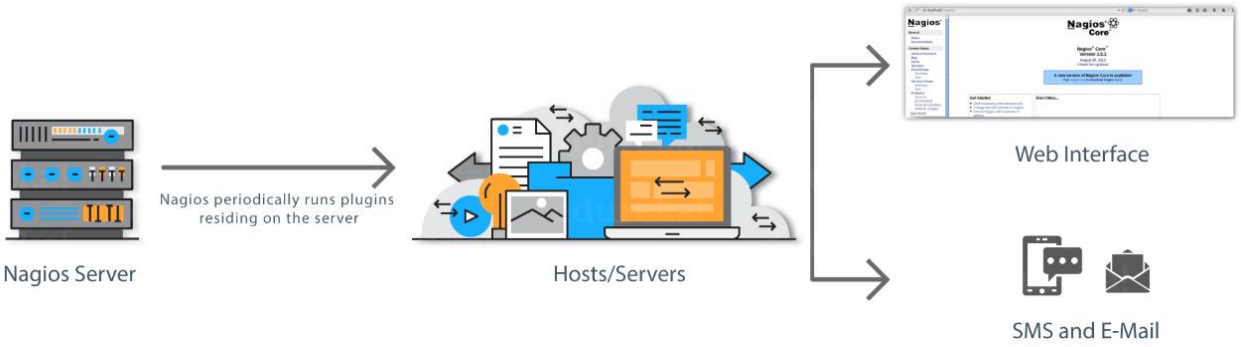
Nagios monitors your entire IT infrastructure to ensure systems, applications, services, and business processes are functioning properly.



For years our security professionals are performing static analysis from – system log, firewall logs, IDS logs, IPS logs etc. But, it did not provide proper analysis and response.

What is Nagios?

Nagios is used for Continuous monitoring of systems, applications, services, and business processes etc in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers. With Nagios, you don't have to explain why an unseen infrastructure outage affect your organization's bottom line.



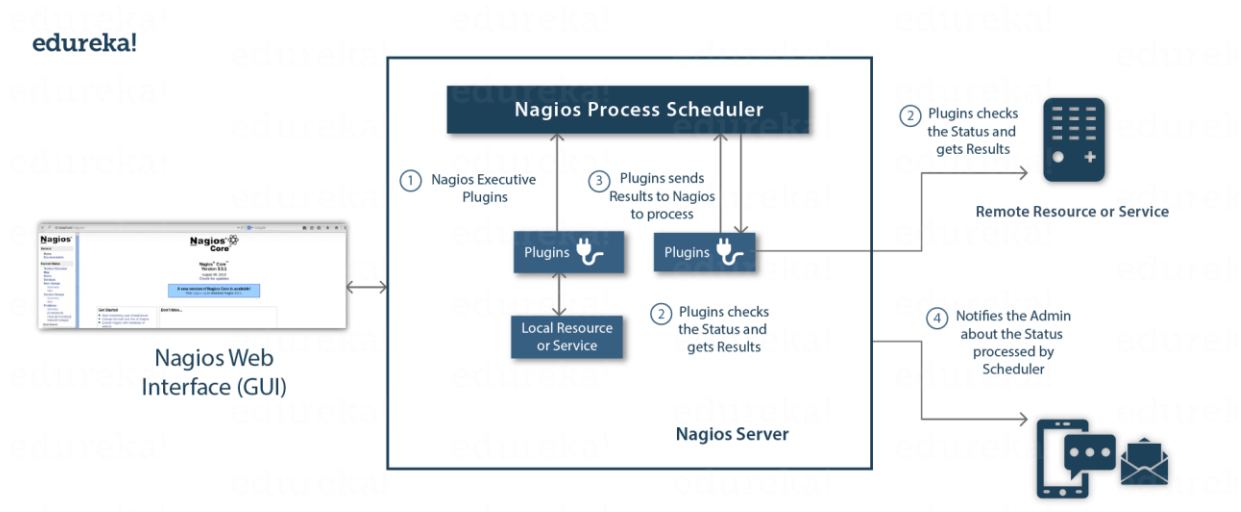
Nagios runs on a server, usually as a daemon or a service.

It periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet. One can view the status information using the web interface. You can also receive email or SMS notifications if something happens.
The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments. It stores the results of those scripts and will run other scripts if these results change.

Plugins: These are compiled executables or scripts (Perl scripts, shell scripts, etc.) that can be run from a command line to check the status or a host or service. Nagios uses the results from the plugins to determine the current status of the hosts and services on your network.

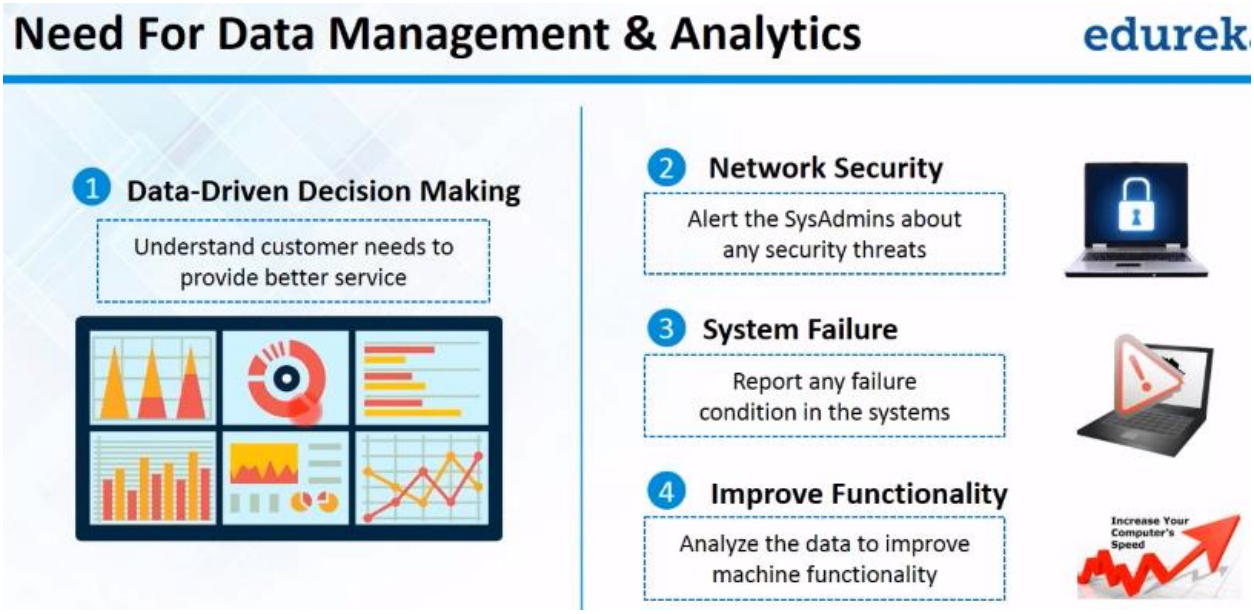
Nagios Architecture

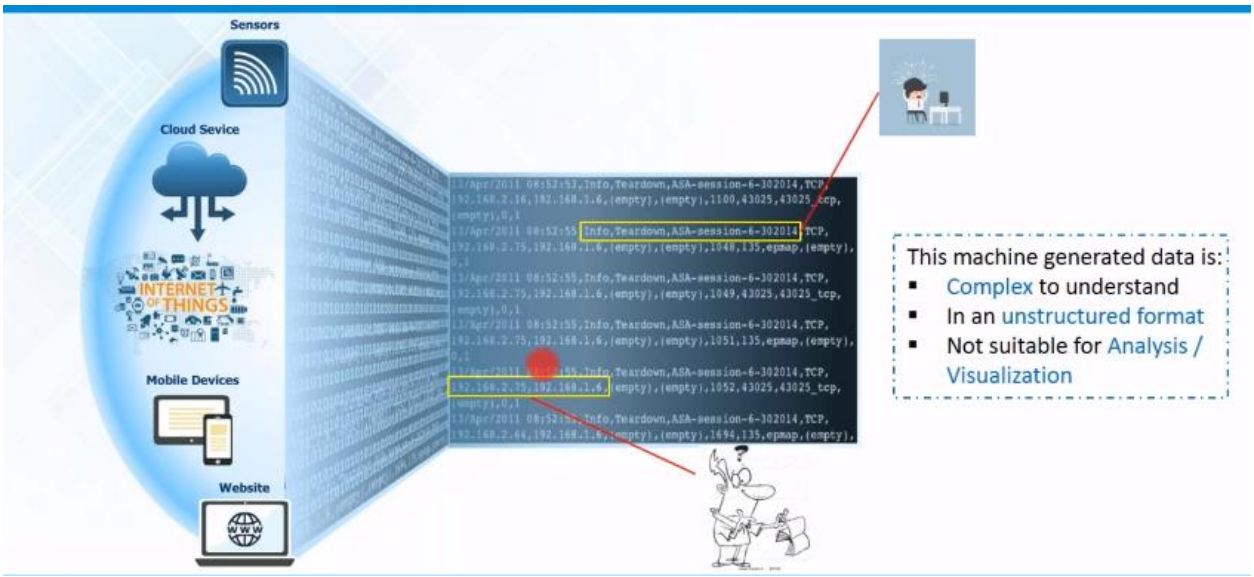
- Nagios is built on a server/agents architecture.
- Usually, on a network, a Nagios server is running on a host, and Plugins interact with local and all the remote hosts that need to be monitored.
- These plugins will send information to the Scheduler, which displays that in a GUI.



14. Splunk

https://www.youtube.com/watch?v=rvjW5LJ0vbU&index=2&list=PL9ooVrP1hQOFPP6mdp1M4Ml436pYZIyH9





Real-Time

Splunk gives you the real-time answers you need to meet customer expectations and business goals.

Machine Data

Use Splunk to connect your machine data and gain insights into opportunities and risks for your business.

Scale

Splunk scales to meet modern data needs — embrace the complexity, get the answers.

AI and Machine Learning

Leverage artificial intelligence (AI) powered by machine learning for actionable and predictive insights.

Splunk Can Be Used To Leverage Machine Data

edureka

1. 

Analyze system performance

2. 

Search & Investigate a particular outcome

3. 

Troubleshoot any failure condition

4. 

Create Dashboards to visualize & analyze results

5. 

Monitor business metrics

6. 

Store and retrieve data for later use

Splunk For Data Analytics

edureka!

splunk>

Splunk is a software platform to search, analyze and visualize the machine-generated data gathered from the websites, applications, sensors, devices etc which make up your IT infrastructure and business.

Advantages Of Using Splunk

- Splunk automatically collects the data in Real-time from multiple systems
- Splunk can accept any data type like .csv, json, log formats, etc
- Splunk can give Alerts / Event notifications
- Splunk satisfies industry needs like Horizontal scalability (using many systems in parallel)
- Splunk can create Knowledge objects for Operational Intelligence

Pull data from multiple systems in real time

Features	Splunk	Sumo Logic	ELK
Searching	✓	✓	Only possible with Integrations
Analysis	✓	✓	Only possible with Integrations
Visualization Dashboard	✓	✓	Only possible with Integrations
SaaS Setup	✓	✓	✓
On Premise Setup	✓	✗	✓
Input any data type	✓	✓	Needs plugins
Plugins & Integration	✓	✓	✓
Customer Support	✓	Available; but not proficient	Available; but not proficient
Documentation & Community	✓	✗	✓

Domino's Use Case

edurek

Interactive Map

- Shows all the orders coming from across US in real time
- Brought employee satisfaction

Real-time Feedback

- Employees constantly see what customers are saying
- Helped them understand customer expectations

Dashboard

- Used to keep score and set targets
- Compare performance with previous week

Payment Process

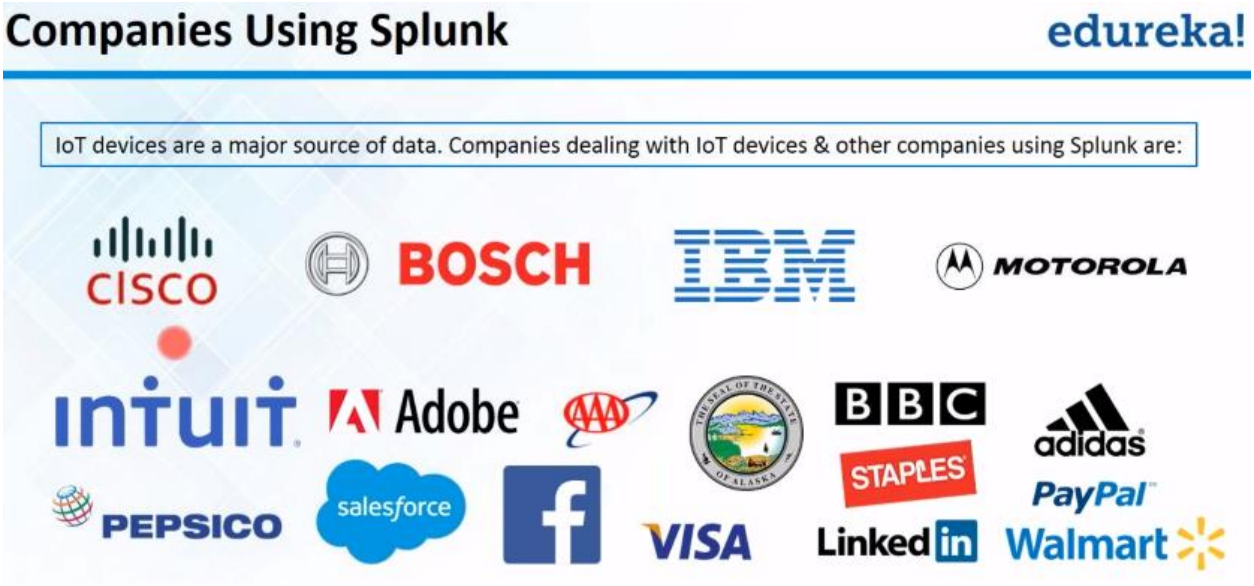
- Analysed the speed of different payment modes
- Determine error free payments modes

Promotional Support

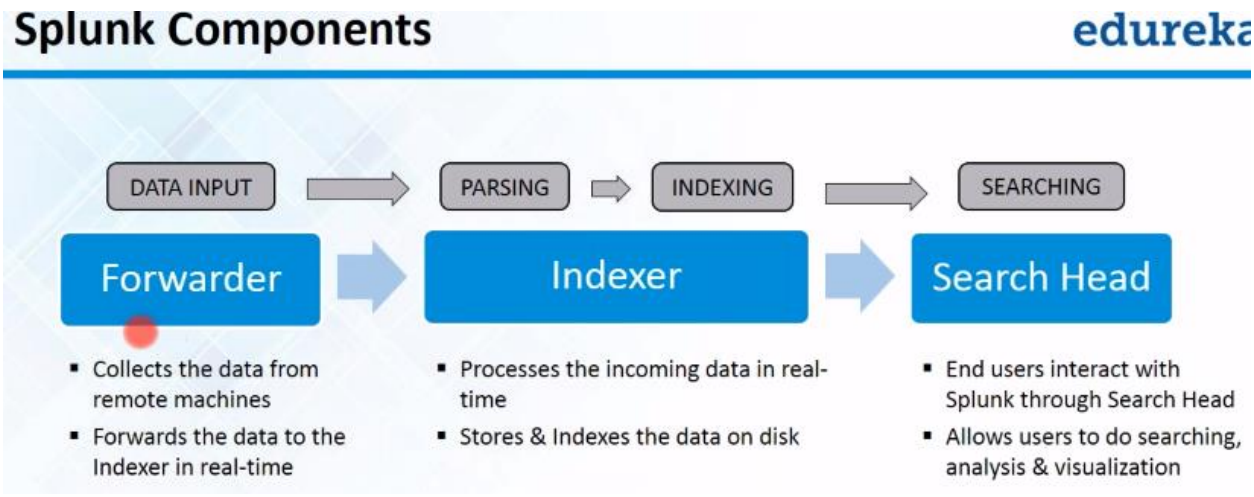
- Track how various promotional offers are impacting in real-time
- Initially, determining the impact of promotions took almost a day

Performance Monitor

- Monitor the performance of Domino's in-house developed point of sales systems

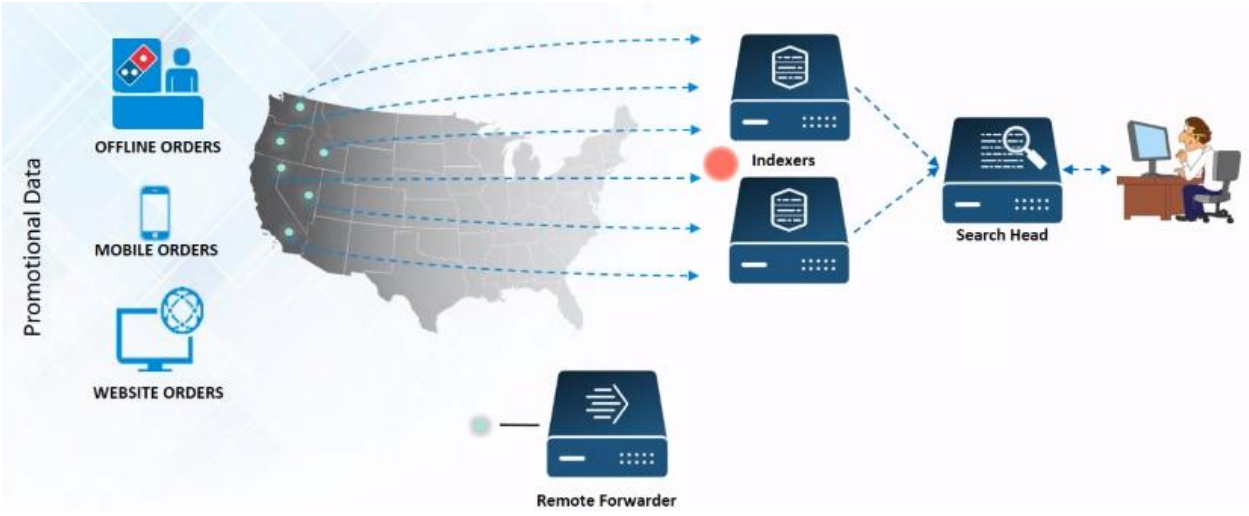

Companies Using Splunk
edureka!
IoT devices are a major source of data. Companies dealing with IoT devices & other companies using Splunk are:
CISCO
BOSCH
IBM
MOTOROLA
intuit
Adobe
BBC
adidas
STAPLES
PayPal
PEPSICO
salesforce
VISA
LinkedIn
Walmart

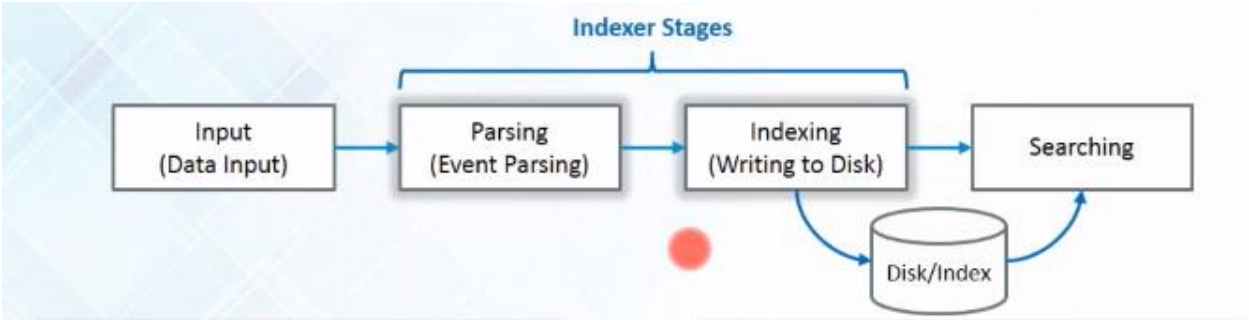
Components


Splunk Components
edureka
DATA INPUT
PARSING
INDEXING
SEARCHING
Forwarder
Indexer
Search Head
Collects the data from remote machines
Forwards the data to the Indexer in real-time
Processes the incoming data in real-time
Stores & Indexes the data on disk
End users interact with Splunk through Search Head
Allows users to do searching, analysis & visualization

Problem Statement

- Dominos had no clear visibility into what offer works the best – in terms of
  - Offer type (for eg 10% off or $2 off)
  - Cultural differences at a region level
  - Device used
  - Time of Purchase
  - Order revenue
- They required insights on consumer behavior and customer response to offers





In the Parsing stage, only relevant data is converted into events:

- Customer Region
- Order revenue
- Time of purchase
- Device used by customers
- Coupons/ Offers used

In the Indexing stage, events are sorted and indexed for storage based on:

- Sales by Geography
- Order revenue
- Time of purchase
- Device preferred by customers
- Coupons/ Offers used

Search Head For Analysis & Visualization

edureka

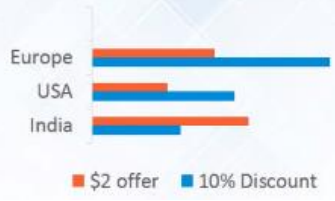
Search Head, is used to gain intelligence and perform reporting.

Dominos used it to get the following insights:

i. Which offer works in which geography?

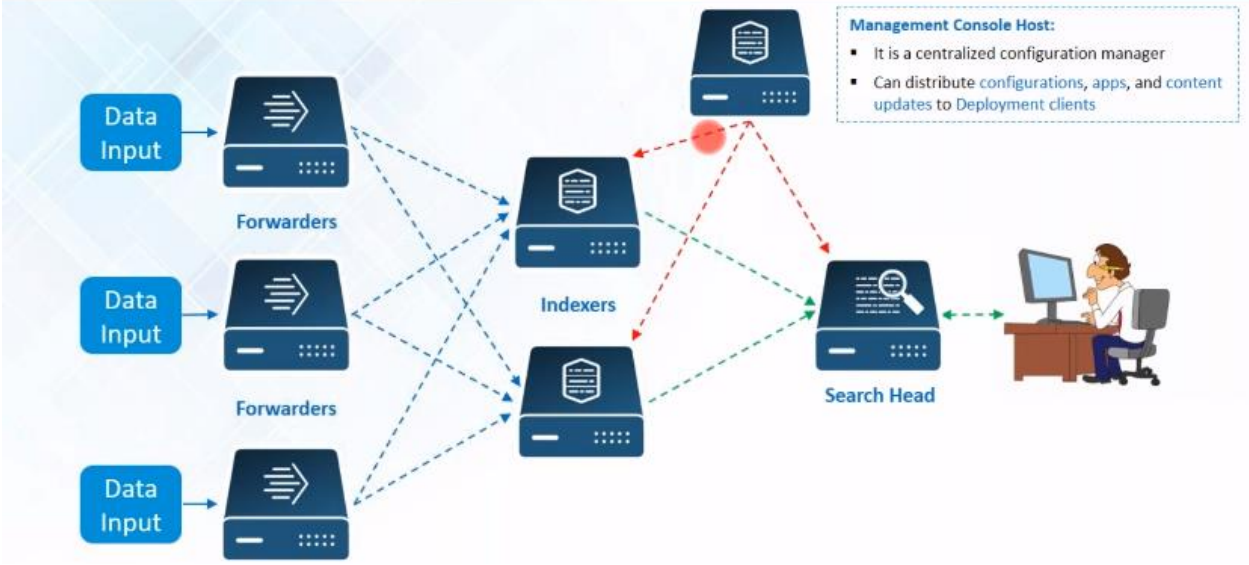
ii. How does the customer behavior change w.r.t changes in order revenue?

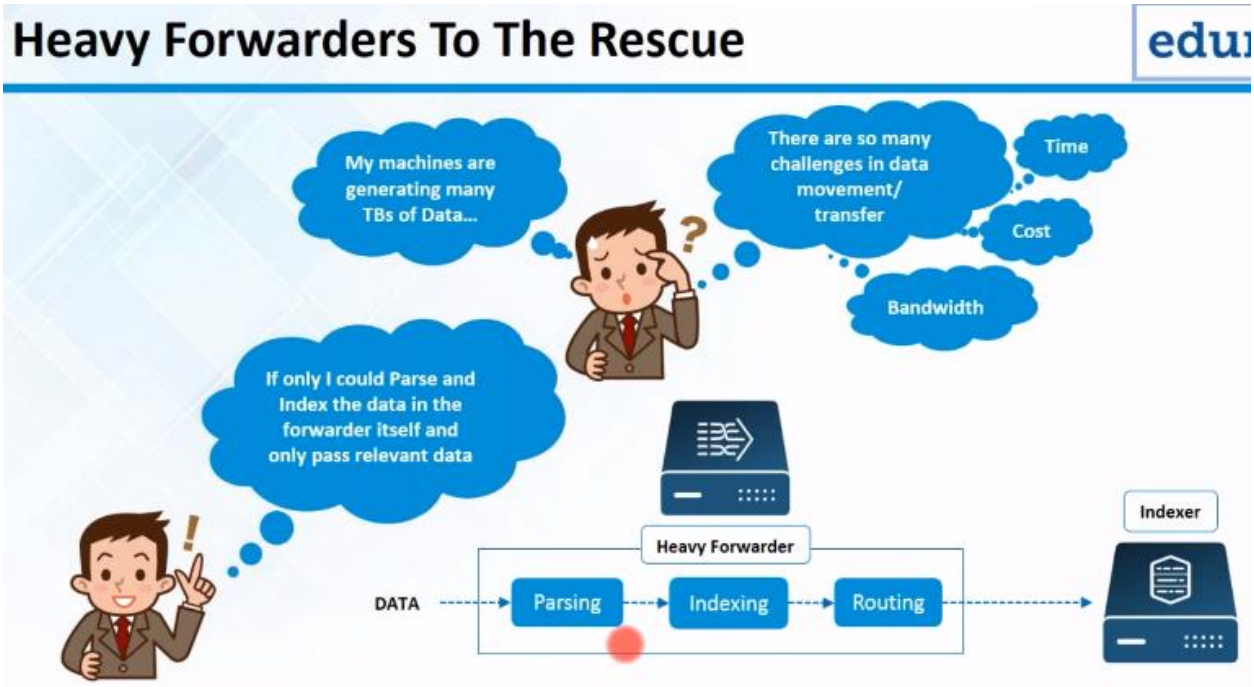
iii. What time of the day is most appropriate for the offers?



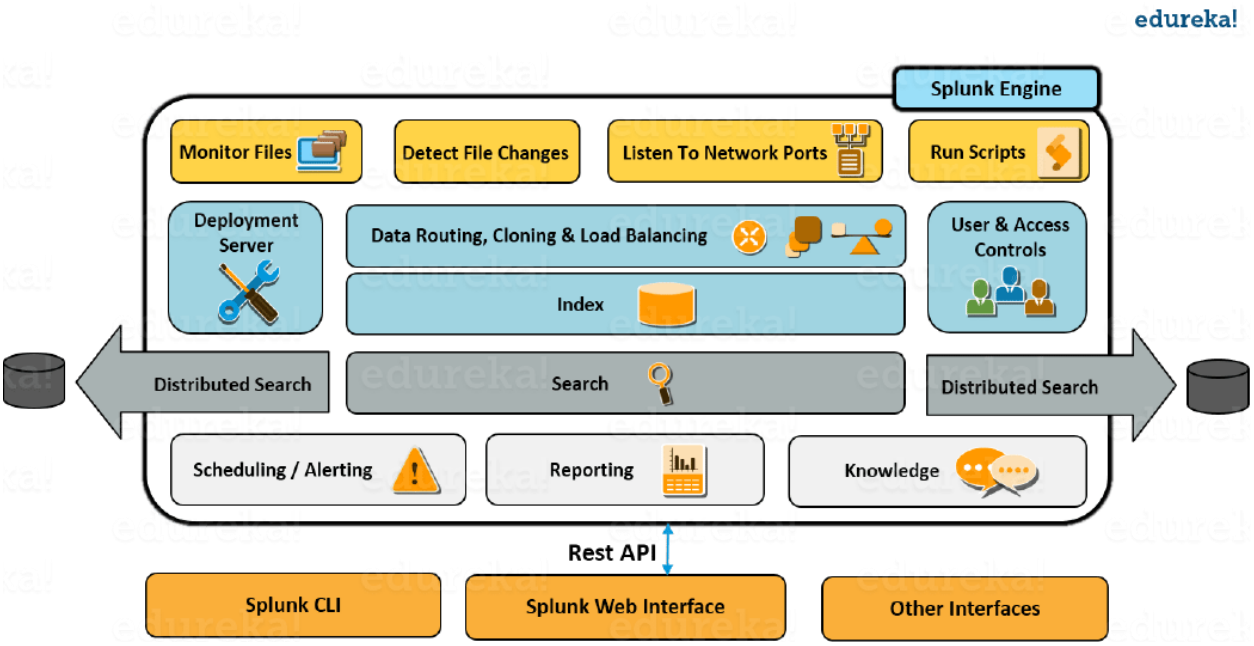
Splunk's End To End Working Pipeline

edureka





Architecture

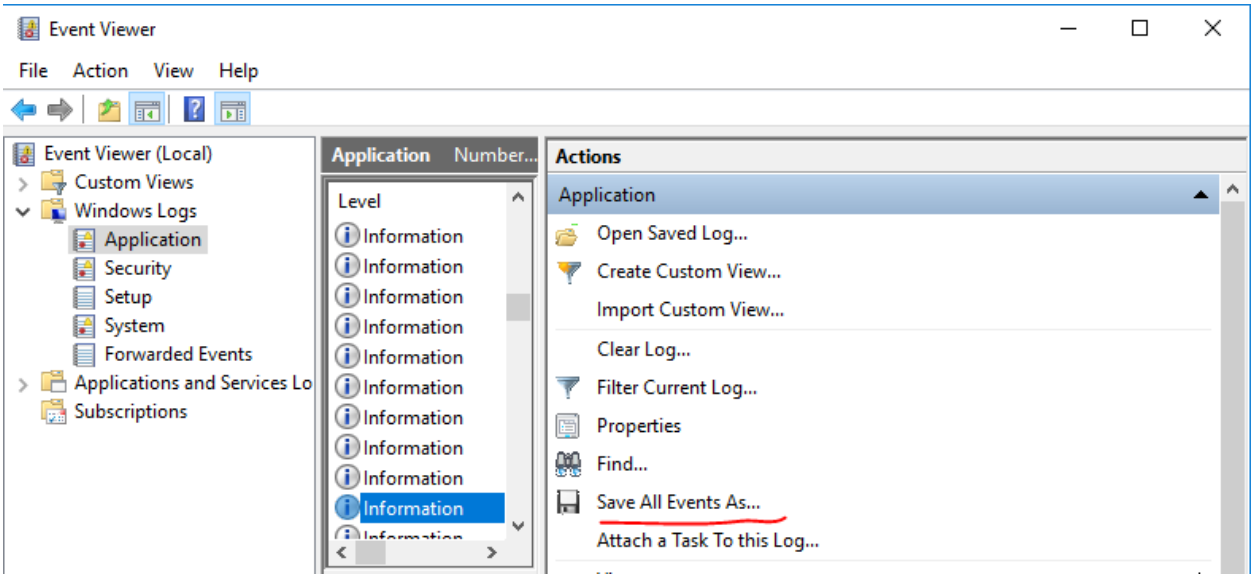


Flavors of Splunk

Enterprise	Cloud	Light

Adding data to Splunk

Windows – Event Viwer



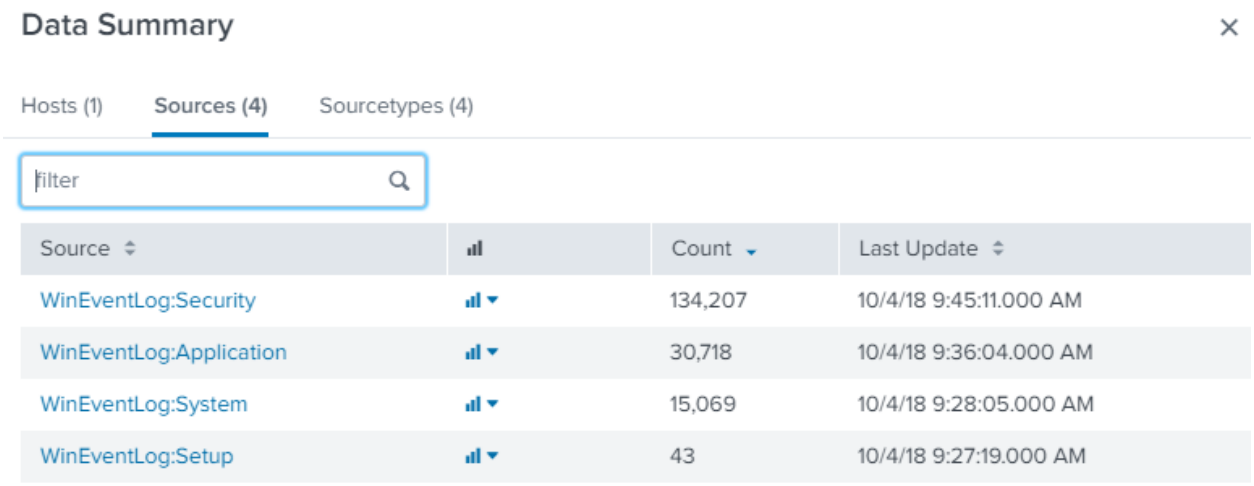
Adding data - Home > Add Data >Monitor > Local Evenet Logs > [Application Security] > Submit

http://localhost:8000/en-US/app/search/search?q=search%20source%3D%22WinEventLog%3A*%22%20host%3D%22HYDPCM457488D%22&earliest=0&latest=&sid=1538625810.13&display.page.search.mode=smart&dispatch.sample_ratio=1&workload_pool=

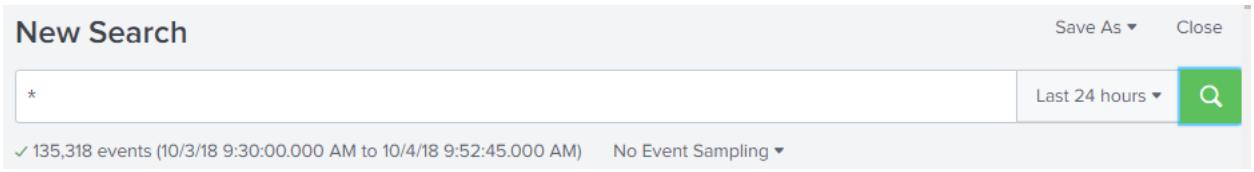
Start Seraching

Go To here : http://localhost:8000/en-US/app/search/search

Check data Summary

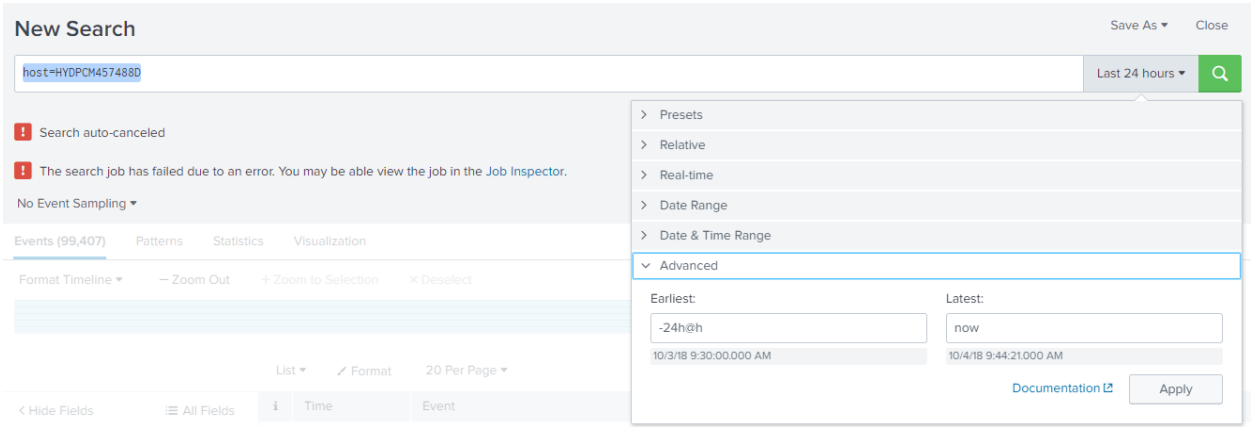


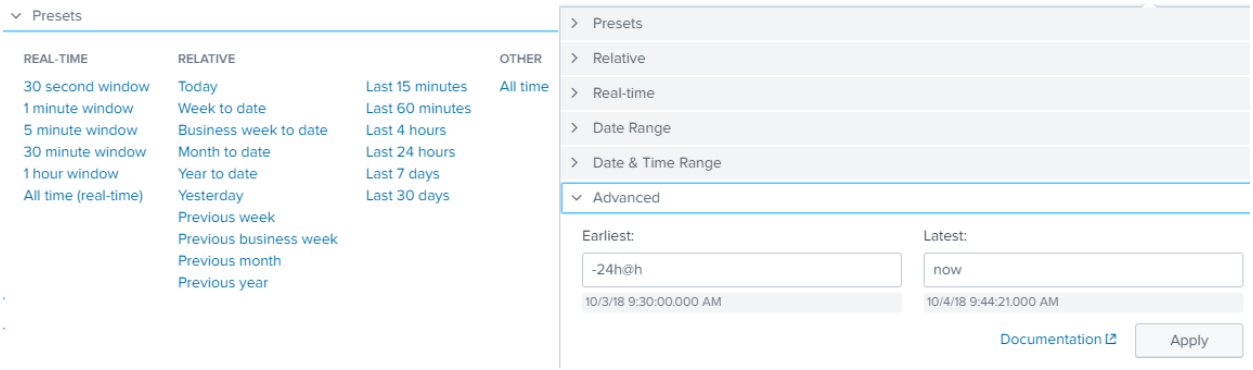
1. "*" - To get all logs



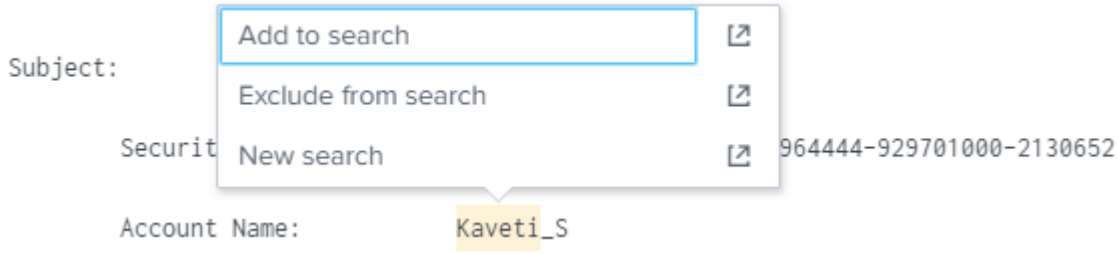
2.host = <Hostname> - To get all logs of perticlular Host/System,

```
host=HYDPCM457488D
```



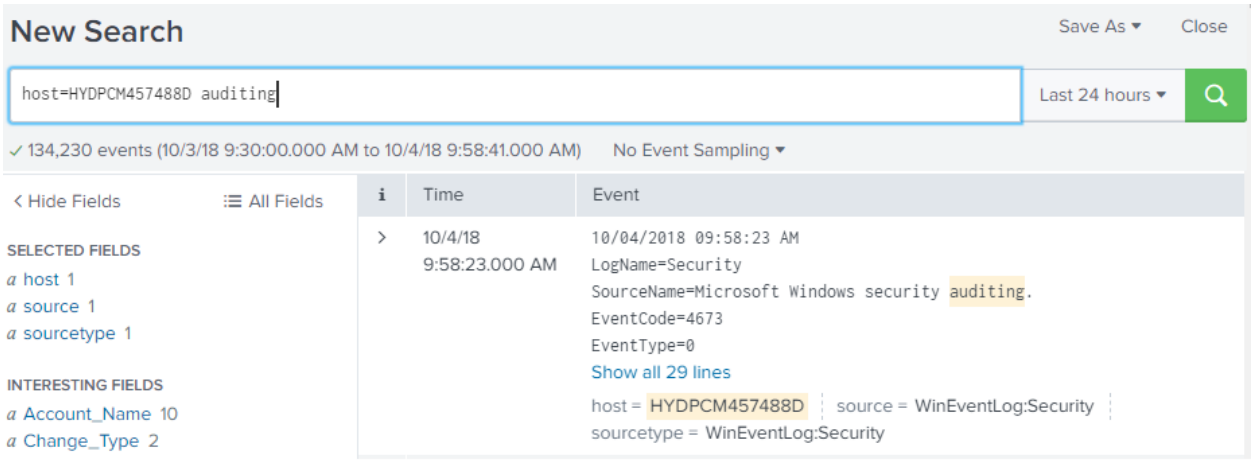


3.Click on Log text & Add to Search



4. To Get the logs which contain the word "`auditing`" in the given host

```
host=HYDPCM457488D auditing
```



SPL – Splunk Processing language

```
source="WinEventLog:*" host="Henson-Lap"
```

Filtering Results

Allows for users to filter results in query. For example show results where event code = 100

```
source="WinEventLog:*" host="Henson-Lap"| search EventCode=100
```

Remove Duplicates

Only shows unique events. For example show only EventCodes once

```
source="WinEventLog:*" host="Henson-Lap"| dedup EventCode
```

```
[query]| head 10
```
◀ Show first 10 results

```
[query]| reverse
```
◀ Reverse result order

```
[query]| sort user
[query]| sort -user
```
◀ Order by user ascending or user descending

```
[query] | tail 10
```
◀ Show last 10 results

1. Get the all the logs of the Host

```
host="hydpcm457488d"
180,149 events (before 10/4/18 10:18:30.000 AM)
```

2.Get the all the Logs of Host=”hyd” & EventCode=5447

```
host="hydpcm457488d" | search EventCode=5447
130,219 events (before 10/4/18 10:19:42.000 AM)
```

3.Get the all the Logs of Host=”hyd” & EventCode=5447 & Process_ID=956

```
host="hydpcm457488d" | search EventCode=5447 | search Process_ID=956
29,594 events (before 10/4/18 10:18:30.000 AM)
```

4.Get the all Logs which conatin tha word `Privileges`

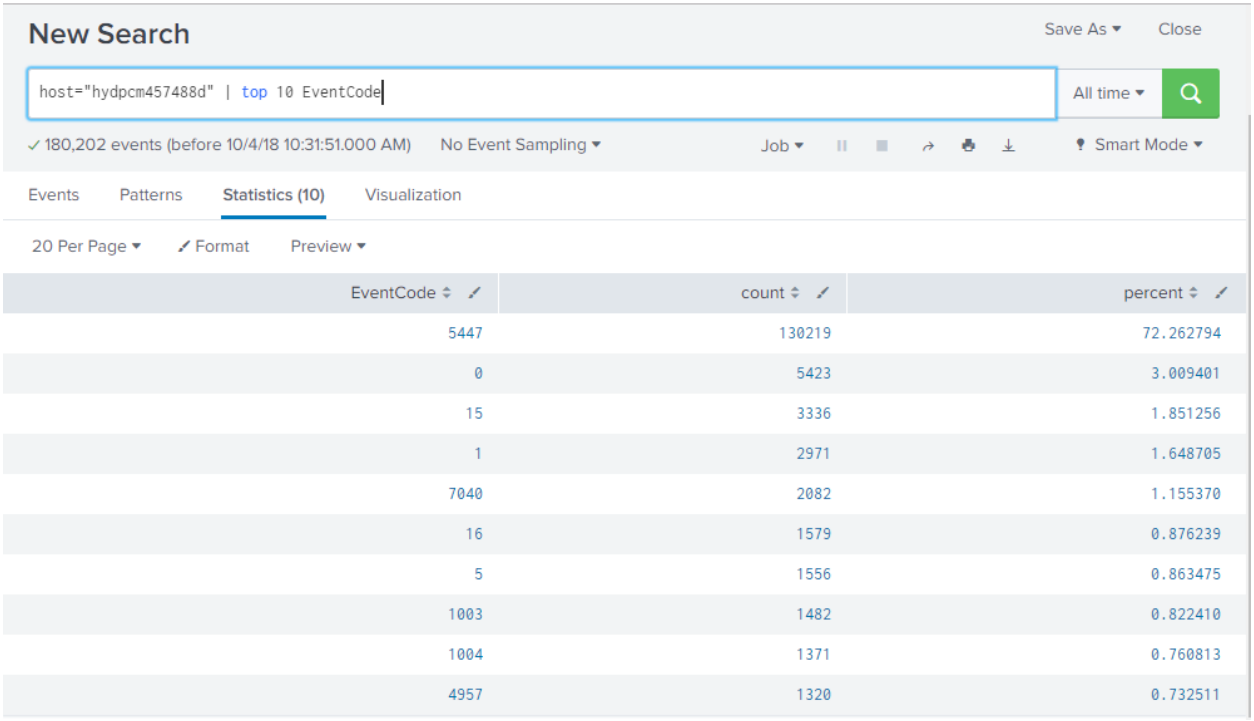
```
host="hydpcm457488d" privileges
374 events     (before 10/4/18 10:38:16.000 AM)
```

```
host="hydpcm457488d" | head 10
10 events (before 10/4/18 10:26:07.000 AM)
```

```
host="hydpcm457488d" | tail 10
10 events (before 10/4/18 10:26:07.000 AM)
```

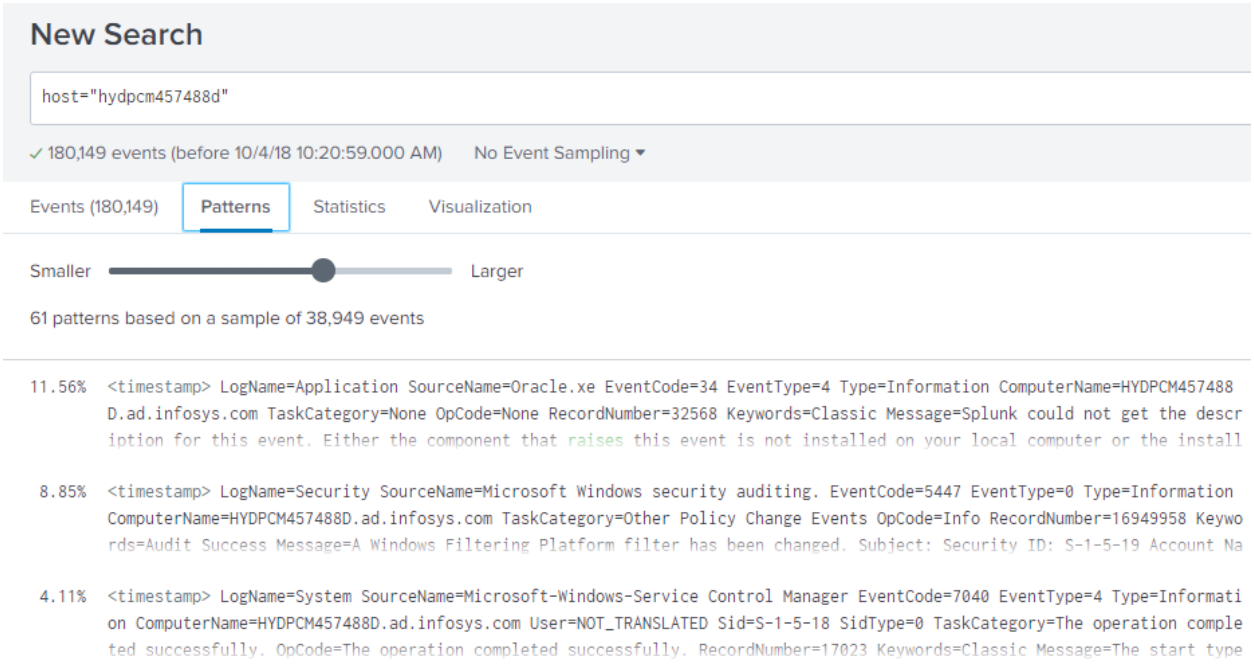
Top Count of EventCodes

```
host="hydpcm457488d" | top 10 EventCode
180,202 events (before 10/4/18 10:31:51.000 AM)
```



EventCode	count	percent
5447	130219	72.262794
0	5423	3.009401
15	3336	1.851256
1	2971	1.648705
7040	2082	1.155370
16	1579	0.876239
5	1556	0.863475
1003	1482	0.822410
1004	1371	0.760813
4957	1320	0.732511

Pattrens – On What conditions/ pattens logs are generated

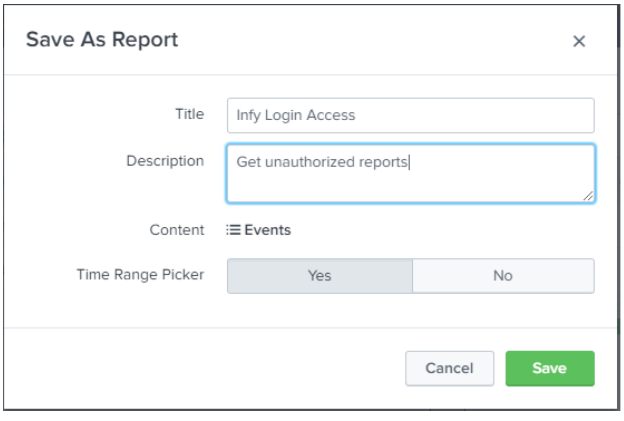


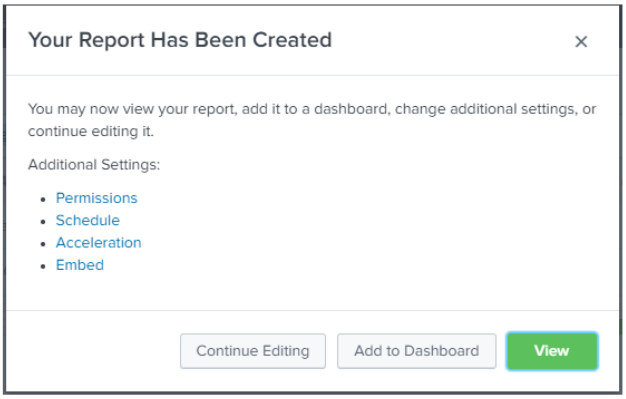
Reporting

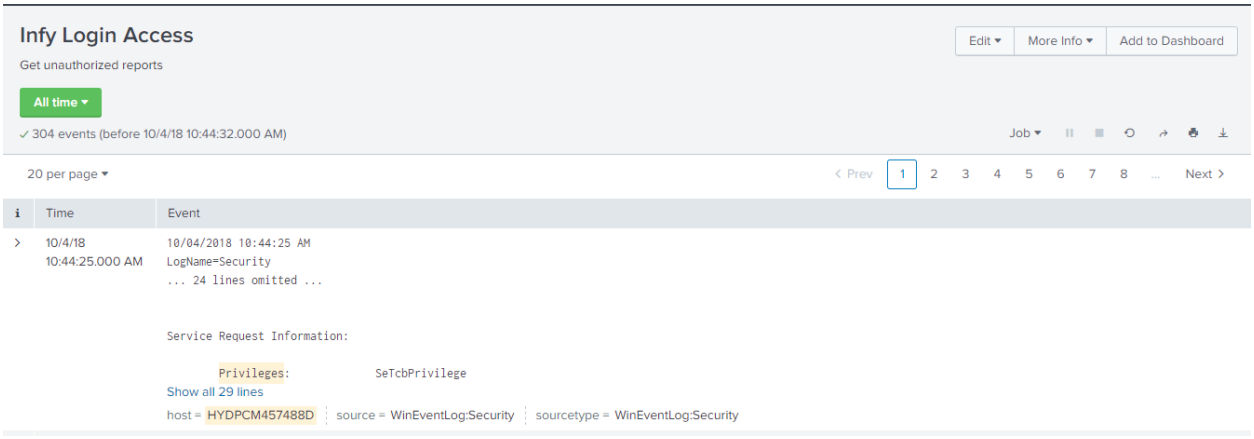
Just I want to get the Reports of UnAuthrozed access of XYZ Company Account.

```
host="hydpcm457488d" privileges Account_Domain=ITLXYZ COMPANY
304 events     (before 10/4/18 10:44:32.000 AM)
```

To Save it as Report : Top on Search > Save As > Report

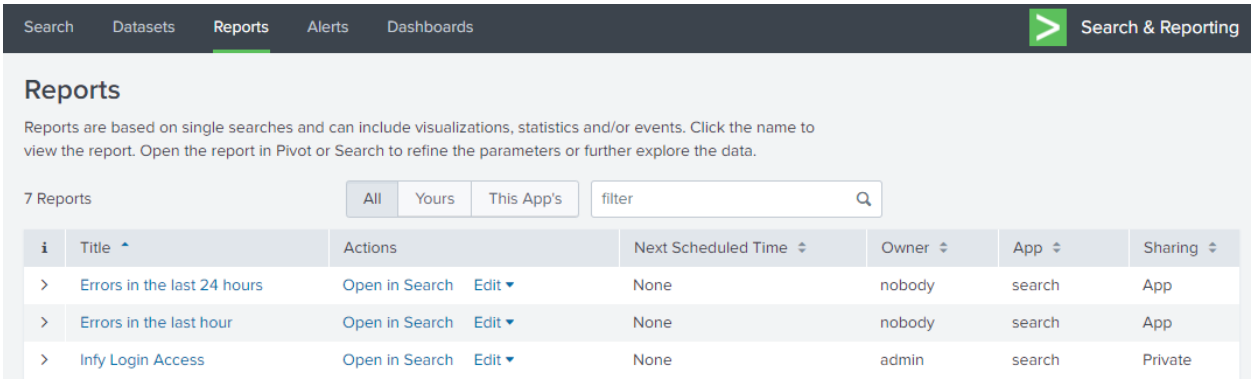






To get Save Reports

Home > Splunk Search > Reports Tab ; we can find the list of reports



Create Alerts

Warnings

- Real-time
- Scheduled

Problems

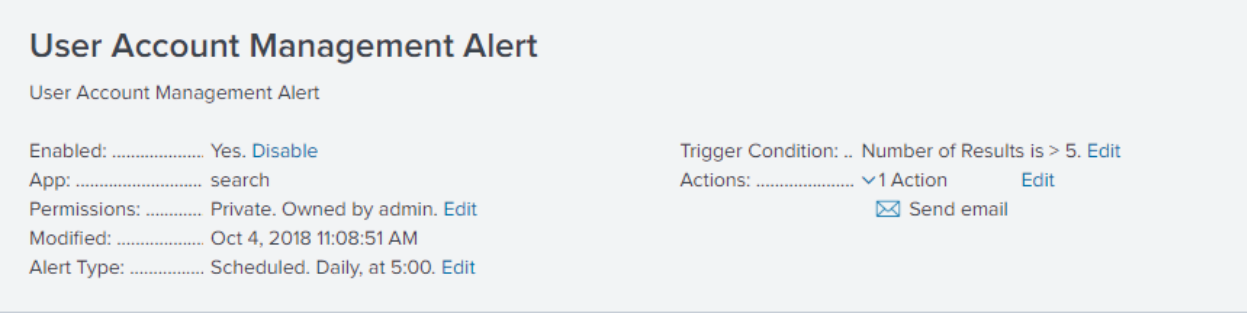
- Quicker resolution
- Actionable

If any one from Infy users , try to modify account settings, should trigger an ALERT!!

```
host="hydpcm457488d" Account management Account_Domain=ITLXYZ COMPANY
```

To Create an Alert,

Search Bar Top > Save As > Alert > Privide Required Details



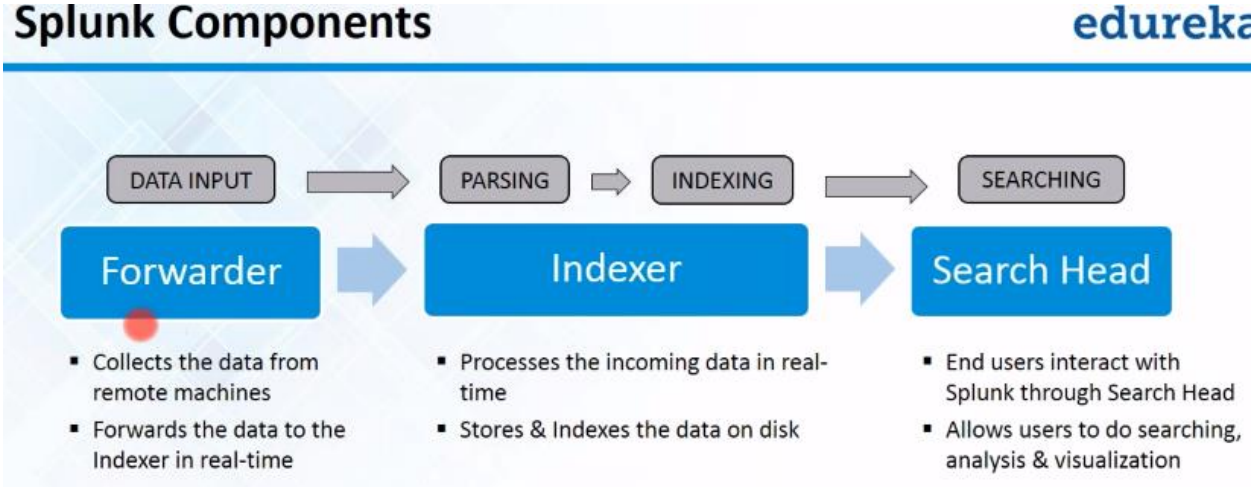
Working with Remote Server Logs

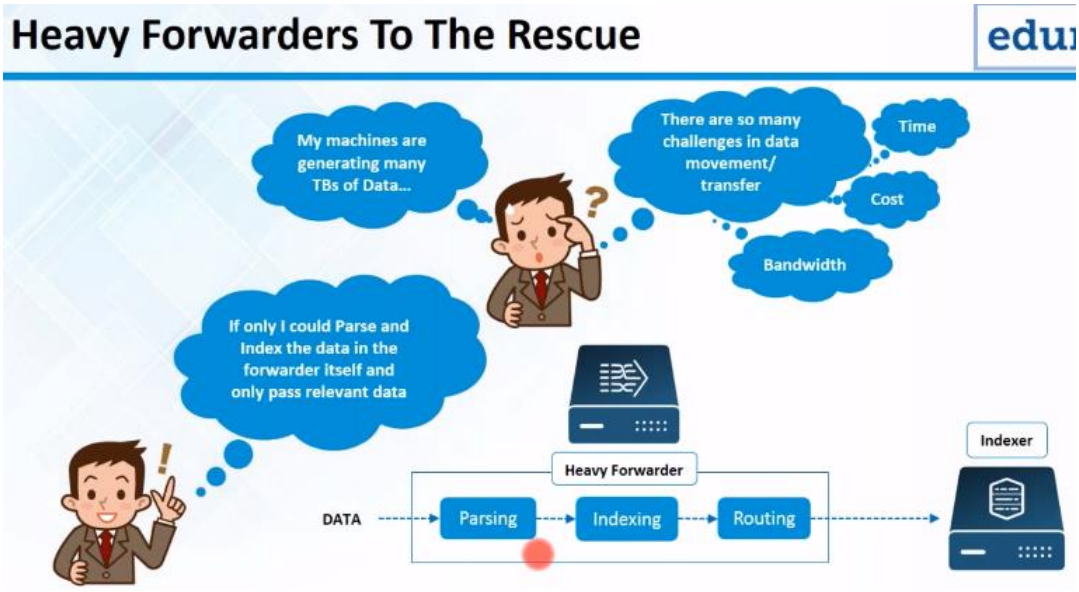
Move logs in Splunk

Different forwarding options

Enterprise architecture

Walk through setting up forwarder





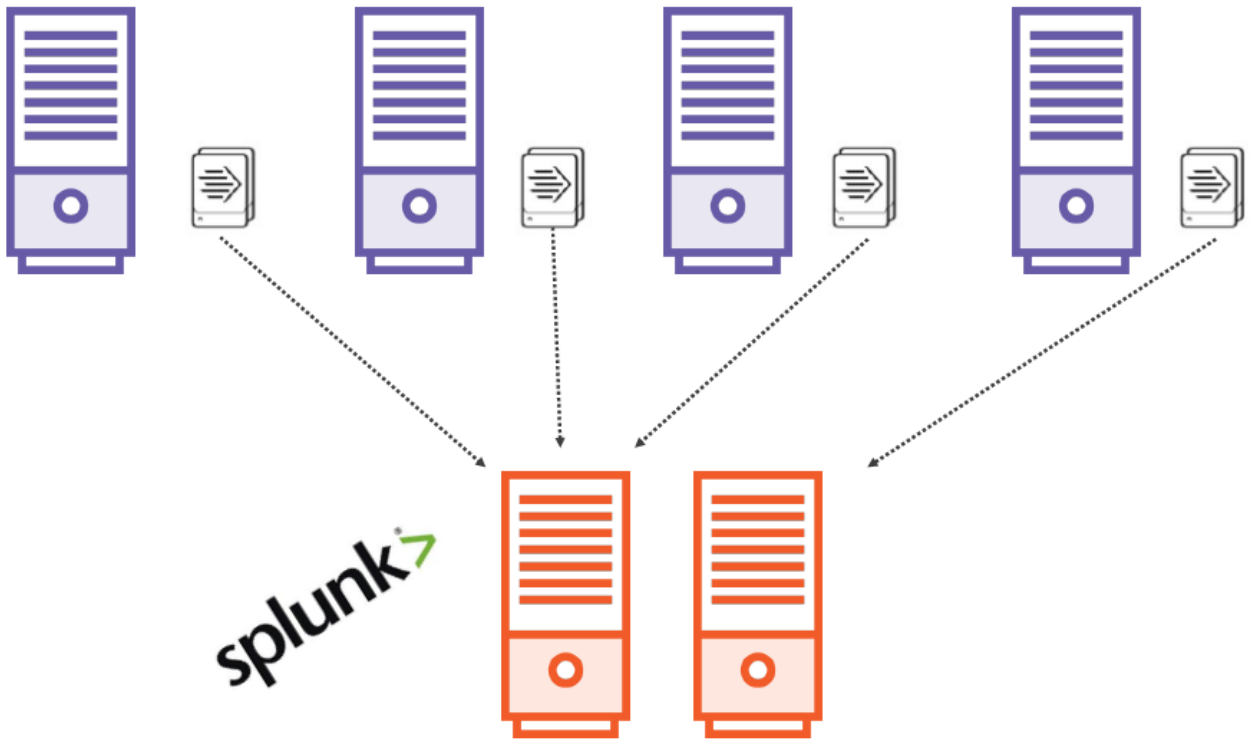
Forwarder

Instance of Splunkthat sends data to another instance of Splunk.

Universal Forwarder	Heavy Forwarder
No Alerts	Full Splunk Instance
No Indexing	Disable Features
Limited Parsing of Data	Web/CLI Configuration
CLI Configuration	

Light forwarder is deprecated as of **Splunk6.0**

Entraprize Splunk Architecuture



Load Balancer : Distributing data across multiple Splunk environments

Setting Splunk Forwarder In Unbuntu

```
www.splunk.com/forwarder
dpkg -i splunk.tgz

cd /opt/splunkforwarder/bin/

sudo ./splunk start

vi config
```

◀ Download Forwarder

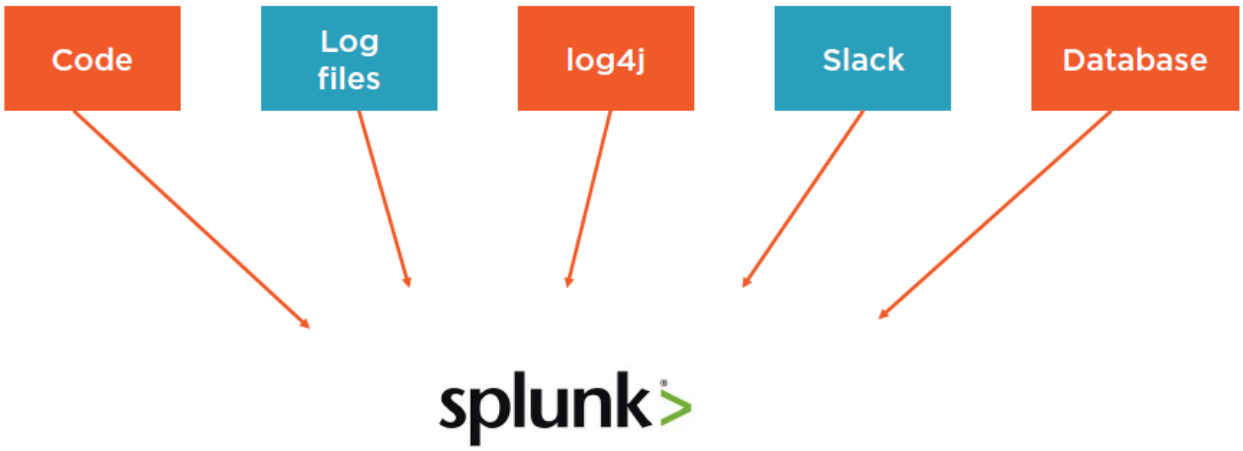
◀ Run install

◀ Change directories

◀ Start Splunk Forwarder

◀ Change Config

Splunk in DevOps



Splunkbase

Splukbase is Market place for Splunkplug-ins and application. Community driven application with licensed and non-licensed options for Splunkapplication.

- Create custom application in Splunk
- Leverage the community for customer "plug-in" Splunk application

Examples of Splunkbase for Tools

- **Microsoft Exchange App**
- **Isilon SplunkApp Splunk**
- **App for Dropbox**

Website : https://splunkbase.splunk.com/apps/

- [ ] Security, Fraud & Compliance
- [ ] IT Operations
- [ ] Utilities
- [ ] Business Analytics
- [ ] IoT & Industrial Data

TECHNOLOGIES >

APP TYPE >

APP CONTENTS >

SPLUNK VERSION >

CIM VERSION >

SPLUNK BUILT & OTHER >

Splunk Add-on for Atlassian JIRA

964 Installs

Splunk App for Jenkins

820 Installs

PagerDuty App for Splunk

812 Installs

Splunk Essentials for Infrastructure

377 Installs

Analytics for Linux

372 Installs

Cisco AnyConnect

334 Installs

Imperva AppSecurity View

218 Installs

OpsGenie for Splunk

216 Installs

Splunk Add-on for New Relic

SolarWinds Add-on for Splunk

Cisco ACI Add-on for Splunk

Splunk App for New Relic

IBM AppScan

IBM® Security AppScan® and Application Security on Cloud enhance web and mobile application security, improve application security program management and strengthen regulatory compliance. Testing web and mobile applications prior to deployment can help you identify security risks, generate reports and fix recommendations.

- **Identify and fix vulnerabilities :** Reduce risk exposure by identifying vulnerabilities early in the software development lifecycle.
- **Maximize remediation efforts:** Classify and prioritize application assets based on business impact and identify high-risk areas.
- **Decrease the likelihood of attacks:**Test applications prior to deployment and for ongoing risk assessment in production environments.

AppScan Document :
http://publibfp.dhe.ibm.com/epubs/pdf/i1328740.pdf?bcsi_scan_510c2960d4f4e50e=0&bcsi_scan_filename=i1328740.pdf

Links

Jenkins : **https://lex.XYZ Companyapps.com/**

Vagrant

Error: Could not resolve host: vagrantcloud.com ?

```
set http_proxy=http://10.219.2.220:80
set https_proxy=http://10.219.2.220:80
```

apt-get update Error ?

```
sudo http_proxy=http://10.219.2.220:80 apt-get update
```