**Ansible**

**Shared libraries:** using groovy and write code and keep in a shared libraries..great way to keep reusable code and edit in 1 place if any vulnerability happens.

**Terraform:** category of infrastructure as code

**Infrastructure as a code have 3 categories:**

1. **Infrastructure provisioning tools**: used for provisioning(creating) infrastructure. Eg :ec2 instance, when u go to cloud (200 diff services) create using provisioning infrastructure.
   * Terraform – multicloud
   * Boto in aws
   * Cloud formation in aws
   * Azure biceps
   * Azure ARM template(json format)
     + Adv: if its is supported in azure its possible
     + maynot be 100 % for biceps and terraform

**Provision means:** create…can modify also..(size)(eg:t2.large to medium)

**Q. can modify? Then y it is called provisioning?**

* It may **delete and recreate**, some times it **shutdown**
* Whenever u make changes in tags, tag is a software, there is no shutdown required because tags are metadata.
* When changing **EC2 instance** type from micro to medium -> **shutdown to restart**-> AWS stops the instance, modifies it, then restarts it.
* In **Azure** -> **delete and recreate**-> because resizing isn't always supported dynamically.
* No guarantee that Changes Will Just Modify because Cloud providers handle resources differently.(aws and azure), Provisioning tools follow cloud provider rules. so they sometimes destroy and recreate resources.
* Most of time when u provision it bring up a new instance.

1. **Configuration management tools:** From 1000 machines upgrade java 11 to 17 and keep everythg else same,

* one way to **delete and create** new 100 machines.this is immutable.chances of corrupt envt is less.
* This ensures a **clean setup with no old configurations**, but:
* There’s **no guarantee** the new machines will work smoothly.
* 90% company prefer **same machine(avoids down time,less risk)**. No confidence of bringing up new machine and work smoothly.
* **Used for upgrading and modifying infrastructure configurations**.
* Manual: people do in diff ways,cannot guarantee the consistency and when error occurs they fix it by their own ways-> delta grows
* At some point of time some code will not work on particular machine. Need to fix without accessing that machine. Give one command and execute it (before they validate and execute). One by one doing without actualy having access to production not good idea. So I can use same script and create new machine. Here high chance of **configuration dripped/snow flakes.**
* BY using ansible ..
  + install ansible on centralized server and connect to the production server.
  + Connection done using ssh ,internally uses scp(**Send configuration files to servers** using scp.),
  + actuall execution happens on these machines,download modules on ansible machine and
  + Execute updates remotely,
  + **Validate before execution** to catch errors before making changes,
  + execute.
  + Ansible
  + Puppet(pull model)
  + Chef(pull model)
* Ansible works **on push pull model**: gather **the facts** and push(configuration changes always push based)
* Others works on pull model
* **Push model**: A **central server** sends (pushes) configurations to target machines.
  + **No agent required**
  + Instant execution from the control node.
  + If the control node is compromised, **attackers can push malicious updates**
* **Pull model**: Each **server (agent) pulls** its configuration from a central server and applies it.
  + **Agents should be installed**
  + Works even if the central server is down temporarily.
  + Requires **agents** to be installed on every server.
* Fact collection is pull model(agent will come and pull and install)

**Q. How Fact Collection Works?**

* + **details (CPU, OS, memory, etc.)** before applying configurations.
  + **In pull model:** Each server runs an **agent** that **pulls configuration** from a central server. The agent collects system facts and applies the correct configuration.
  + **Push model**: Ansible collects facts from a central server and pushes configurations to target machines via **SSH**.
* Agent on the production server if we are using puppet, agent will pull the configurations and execute
* Push model like central person, don’t need an agent (ssh)
* Push model is vulnerable..
  + If an attacker **gains access to the Ansible control node**, they can **push malicious updates** to all servers.

1. **Server templating tools:** Server templating tools help create pre-configured environments so you can quickly set up and manage servers without manual setup.
   * Vagrant (help to create vm)
   * Docker (create containers)