Infrastructure as Code

Iron age vs Cloud age

- In the “iron age”, provisioning and maintaining infrastructure was manual work.
- In the “iron age”, focus was more on manual process, time consuming change management process because getting it wrong was expensive.
- In the “cloud age”, systems are decoupled from the physical hardware.
- In the “cloud age”, all routine work like provisioning, changes can be done in minutes.

Why Infrastructure as Code?

- Provision, configure, update and maintain infrastructure and services in less time.
- Quickly detect and resolve the infrastructure and services issues.
- Systems should be consistently configured and up to date.
- Infrastructure team should spend less time on routine work and focus on infrastructure improvements to enable organization to meet the ever-changing needs of the business.
- In fact, cloud and automation often makes things worse.
- Adopting cloud and automation tools immediately lowers the barriers for making changes to the infrastructure but managing changes in a way that improves consistency and reliability does not come out of the box with any tool. You have to think through about tools, systems, processes and habits (culture of the team) to use them effectively.

What is Infrastructure as Code?

- Infrastructure as code is an approach to infrastructure automation based on practices from software development.
- It emphasizes consistent, repeatable routines for managing the infrastructure.
- Treat your infrastructure as Software.
- Infrastructure as code is not just an automation.

Goals of Infrastructure as Code.

- Infrastructure team supports and enables change, rather than being an obstacle or a constraint.

- Changes to the system are routine, without drama or stress.

- Infrastructure team should spend their time on improving automation, infrastructure reliability, not on routine and repetitive tasks.

- Users are able to define, provision and manage the infrastructure resources they need, without needing Infrastructure team to do it for them.
- Improvements are made continuously.

Challenges with Dynamic Infrastructure

- Server Sprawl
- Configuration Drift
- Snowflake Servers.
- Fragile Infrastructure.
- Automation Fear.
- Erosion.

Principles of Infrastructure as Code

- Systems can be easily reproduced.
- Systems are consistent.
- Processes are repeatable.
- Design is always changing.

Practices of Infrastructure as Code

- Use definition files to define your infrastructure.

- Self documented systems and processes.

- Keep documentation close to your code.

- Version all the things.
    - Traceability
    - Rollback
    - Correlation
    - Visibility
    - Actionability

Tools

- Terraform - Orchestration Engine
- Saltstack - Configuration Management System and Event Driven Infra Management
- Kubernetes - Container Orchestration
- Jenkins - CI & CD (pipeline)
- Spinnaker - CD (POC will start soon)
- Datadog - Monitoring
- Packer - AMI Management
- Katello - RPM Repository Management
- Bitbucket - Version Control System
- RackHD for h/w provisioning.
- Chatbots using Slack (POC will start soon)

Questions?