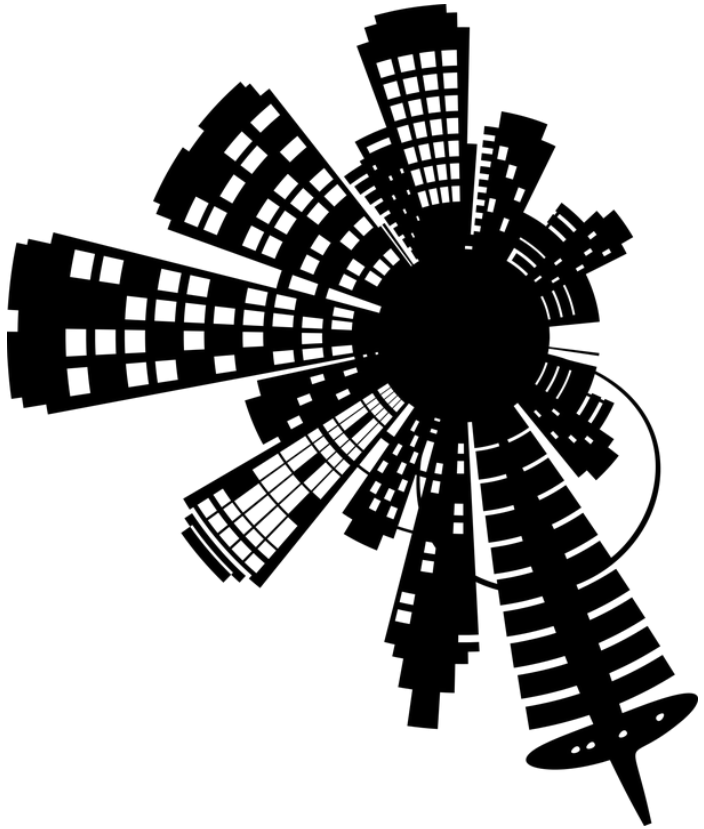
SACON International 2017

India | Bangalore | November 10 – 11 | Hotel Lalit Ashok



Immutable Infrastructure

Nilanjan De
FireCompass, Inc.
nde@firecompass.com

Agenda

- What is Immutable Infrastructure?
- Advantages / Drawbacks
- How to?
- Demos

What is Immutable Infrastructure?

- Aka Immutable servers, phoenix servers
- Traditionally
  - “Snow-flake” Servers are continually updated and modified in place
  - SSH into the servers, update, patch, tweak config, deploy new code
- An *immutable infrastructure* is another infrastructure paradigm in which servers are never modified after they're deployed.
- Classic Immutability
  - Linux Live CDs

Rules for Truly Immutable Servers

- Rule 1:
  - Don’t change the OS filesystem
  - Don’t install / upgrade / downgrade / remove packages
  - Not even for security vulnerabilities!
  - Don’t edit configuration
  - No hot-fixes to your app code
    - Not even small or urgent fixes
- Rule 2:
  - If tempted to do change anything
    - Follow Rule 1

Semi-immutable servers

- Only automated system updates allowed
- No logins / SSH
- No manual changes

Phoenix Servers

- Immutable servers with a TTL (Time to live)
- Servers Self-destruct after e.g. 1 week
- Recreate servers regularly

How do we upgrade?

- Create updated server from scratch
  - Updated OS
  - Updated App
  - Updated Configuration
- Deploy
- Validate / test the deployment
- Snapshot the image and create multiple servers if required
- Redirect production traffic to new deployment
- Shutdown old servers or optionally keep them around, just in case

Advantages – Security Benefits

- Reduced attack surface
  - Hardened OS
  - Remove packages not required including SSH, bash, wget
- Attacker thrown out with old server
  - Backdoors installed by attackers do not persist
  - They may still get back if vulnerabilities are not patched
- Better Auditing and Monitoring
  - Centralized monitoring
  - Log unexpected file changes
  - Log unexpected network connections

Advantages

- Defined by code, hence version-able
- Relatively easy to replicate to a point in time
- Easier to scalable ‘on-demand’
- More consistent Staging environments

Drawbacks

- Small changes are cumbersome
  - Solution: Automation
- Debugging / Forensics are difficult
  - Solution: Bundle debugging tools, ship logging/debugging output to centralized log servers (e.g., ELK)
- Some applications / components, e.g., databases or anything stateful may not fit well into this paradigm
  - Re-architect your application / components
  - Implement database refactoring – separate database updates from application updates.
  - Pass the buck to something like Amazon RDS which maintains the database or use AWS EBS to persist the data

Drawbacks

- May increase cost if you plan to keep old phoenix servers.
  - Solution: Delete old servers automatically
- DevOps maturity is required to ensure that you can roll out an update quickly (e.g., in case of a 0day flaw)

How to implement? Recommendations

- Servers in any virtualized environment (like containers, preferably in a cloud computing environment)
  - Isolated instances
  - Fast Provisioning from custom images
- Full automation of your deployment pipeline
- A service-oriented architecture
- Stateless, Volatile Application layer
- Persistent Data Layer
  - External Centralized logging – ELK
  - External data sources – AWS RDS, Data volumes
- Dedicated DevOps team

Demo

- Immutable containers using docker
- https://docs.docker.com/get-started/

Relevant Tools & Technologies

- Continuous Delivery Platform
  - Spinnaker, Jenkins, GoCD
- Cloud platforms
  - AWS, GCP, Azure, Kubernetes
- Containers
  - Docker, docker-compose, AWS ECS, Google Container engine
- Cloud functions – Serverless Architectures
  - AWS Lambda, Google cloud functions
- Netflix’s Chaos Monkey

Further Reading

- https://www.digitalocean.com/community/tutorials/what-is-immutable-infrastructure
- https://www.thoughtworks.com/insights/blog/moving-to-phoenix-server-pattern-introduction
- https://martinfowler.com/bliki/ImmutableServer.html
- https://devops.stackexchange.com/questions/49/what-are-the-pro-and-cons-of-snowflakes-servers-phoenix-servers-and-immutable-s
- https://www.slideshare.net/jpetazzo/immutable-infrastructure-with-docker-and-containers-gluecon-2015
- https://www.slideshare.net/SonatypeCorp/there-is-no-server-immutable-infrastructure-and-serverless-architecture