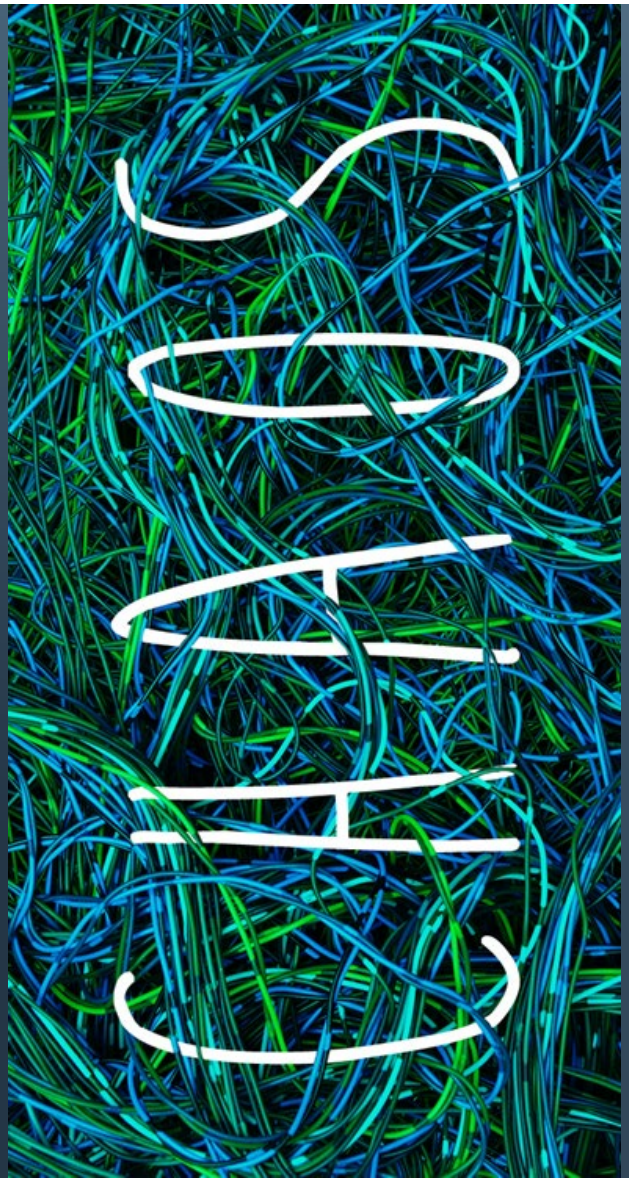
Booz | Allen | Hamilton



EMBRACING CHAOS

By: Onik Quddus

FEBRUARY 2019

AGENDA

THE PROBLEM

CHAOS ENGINEERING

DEVSECOPS

NETFLIX

SIMIAN ARMY

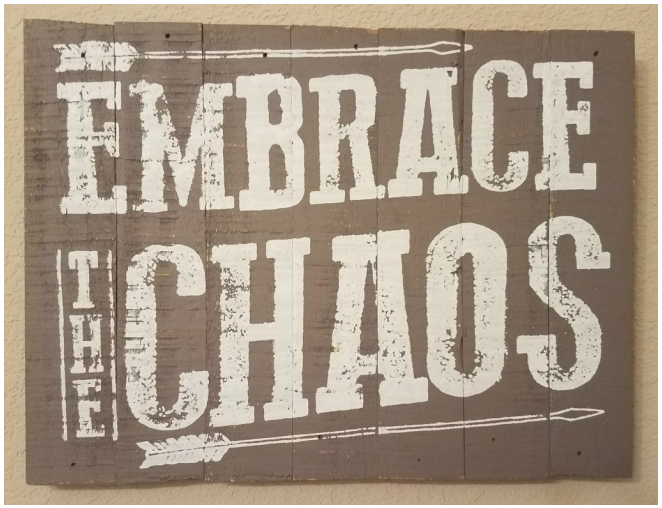
CHAOS MONKEY

REFERENCE ARCHITECTURE

CLOUD RESILIENCY BEST PRACTICES

THE PROBLEM

- Critical mission applications and services need to be resilient, highly available, redundant, and efficient
- Unpredictable outcomes compounded by rare but disruptive events can drastically affect production environments
- Failure is not an option, though it is inevitable
- Best way to prevent failure is to embrace the chaos to ensure success

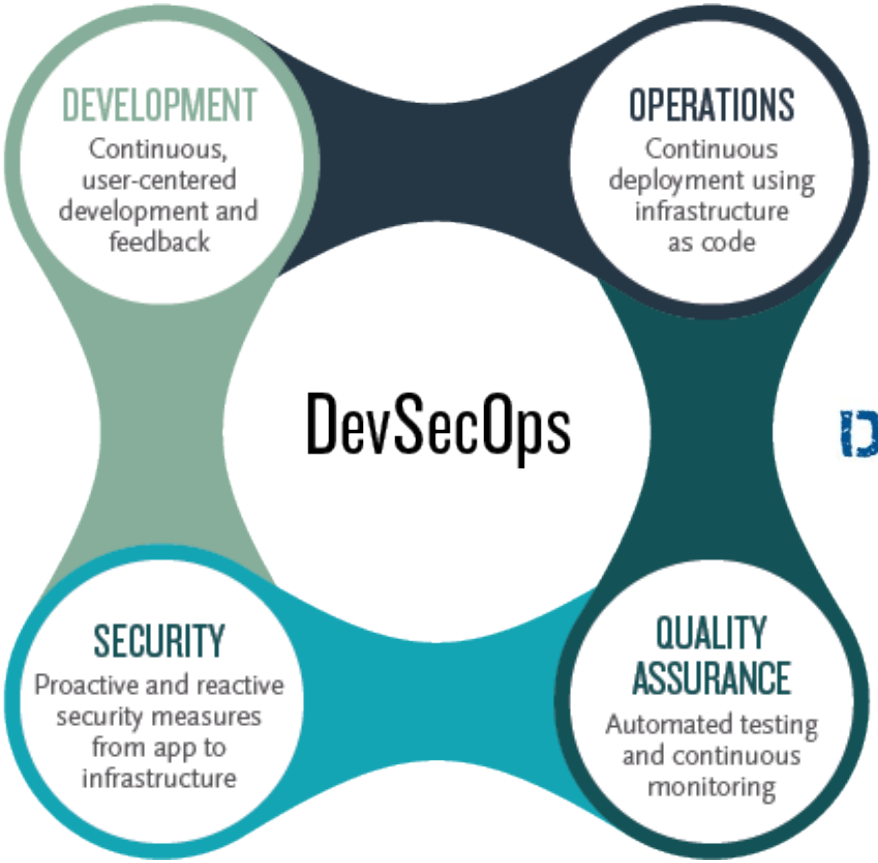


CHAOS ENGINEERING

THE DISCIPLINE OF EXPERIMENTING ON A DISTRIBUTED SYSTEM IN ORDER TO BUILD CONFIDENCE IN THE SYSTEM'S CAPABILITY TO WITHSTAND TURBULENT CONDITIONS IN PRODUCTION

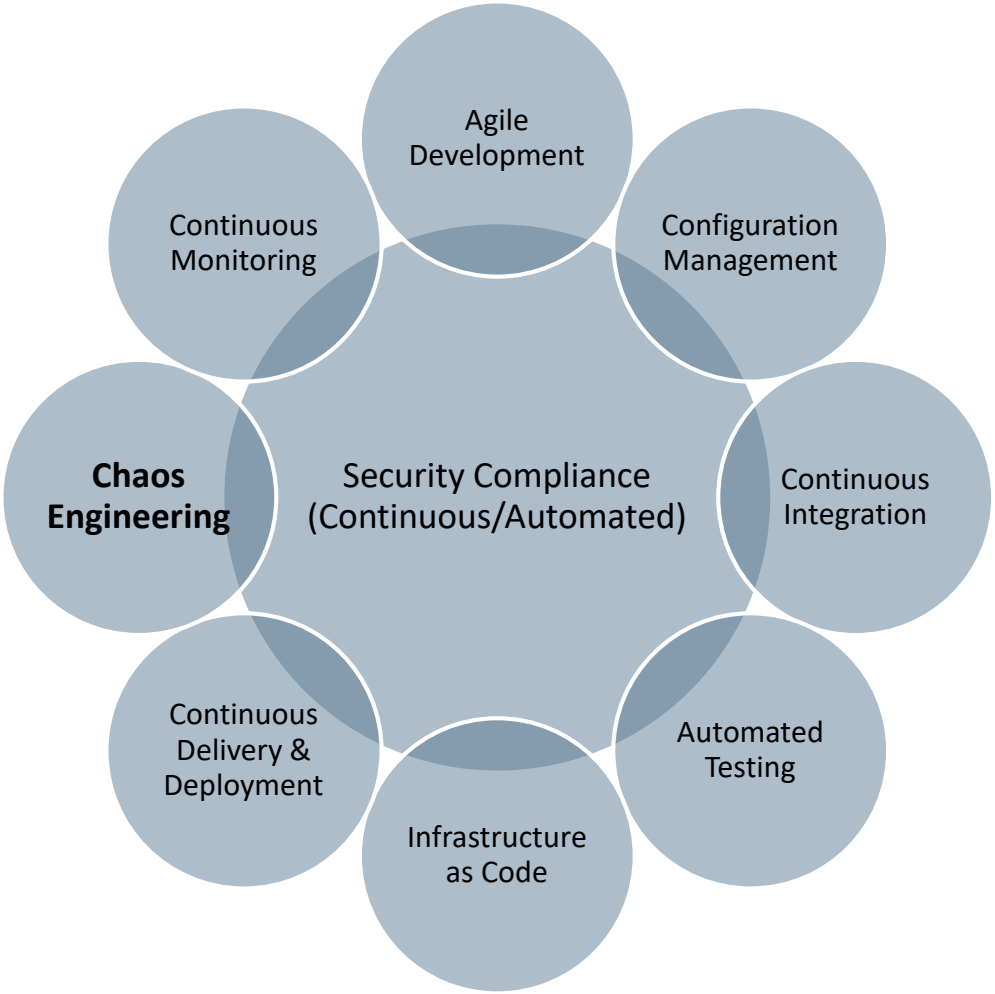
- Way to increase confidence- Consistent Process, Unit Tests, Code Coverage, Automation test only the known
- The Principles of Chaos provide confidence to innovate quickly to meet the speed of need, while accounting for the unknown
- Chaos in Practice
  - Build a Hypothesis around Steady State Behavior
  - Vary Real-world Events
  - Run Experiments in Test/Production
  - Automate Experiments to Run Continuously
  - Minimize Blast Radius

DEVSECOPS



DEVSECOPS | SECURITY AS CODE

DEVSECOPS BEST PRACTICES



NETFLIX CHAOS ENGINEERING

- Netflix developed the open source product called Simian Army during their migration to the cloud
- Netflix set out to build a system designed to accommodate failure at any level
- Philosophy: “The best way to avoid failure is to fail constantly”
- Created the Simian Army tool suit to test and verify resiliency of cloud based systems

SIMIAN ARMY

- **Chaos Monkey-** Kills random instances and clusters
- **Chaos Gorilla-** Kills entire zones
- **Chaos Kong-** Kills entire regions
- **Latency Monkey-** Degrades network and injects faults
- **Conformity Monkey-** Looks for outliers
- **Circus Monkey-** Kills and launches instances to maintain zone balance
- **Doctor Monkey-** Fixes unhealthy resources
- **Janitor Monkey-** Clean up unused resources
- **Howler Monkey-** Yells about bad things like AWS limit violations
- **Security Monkey-** Finds security issues and expiring certifications

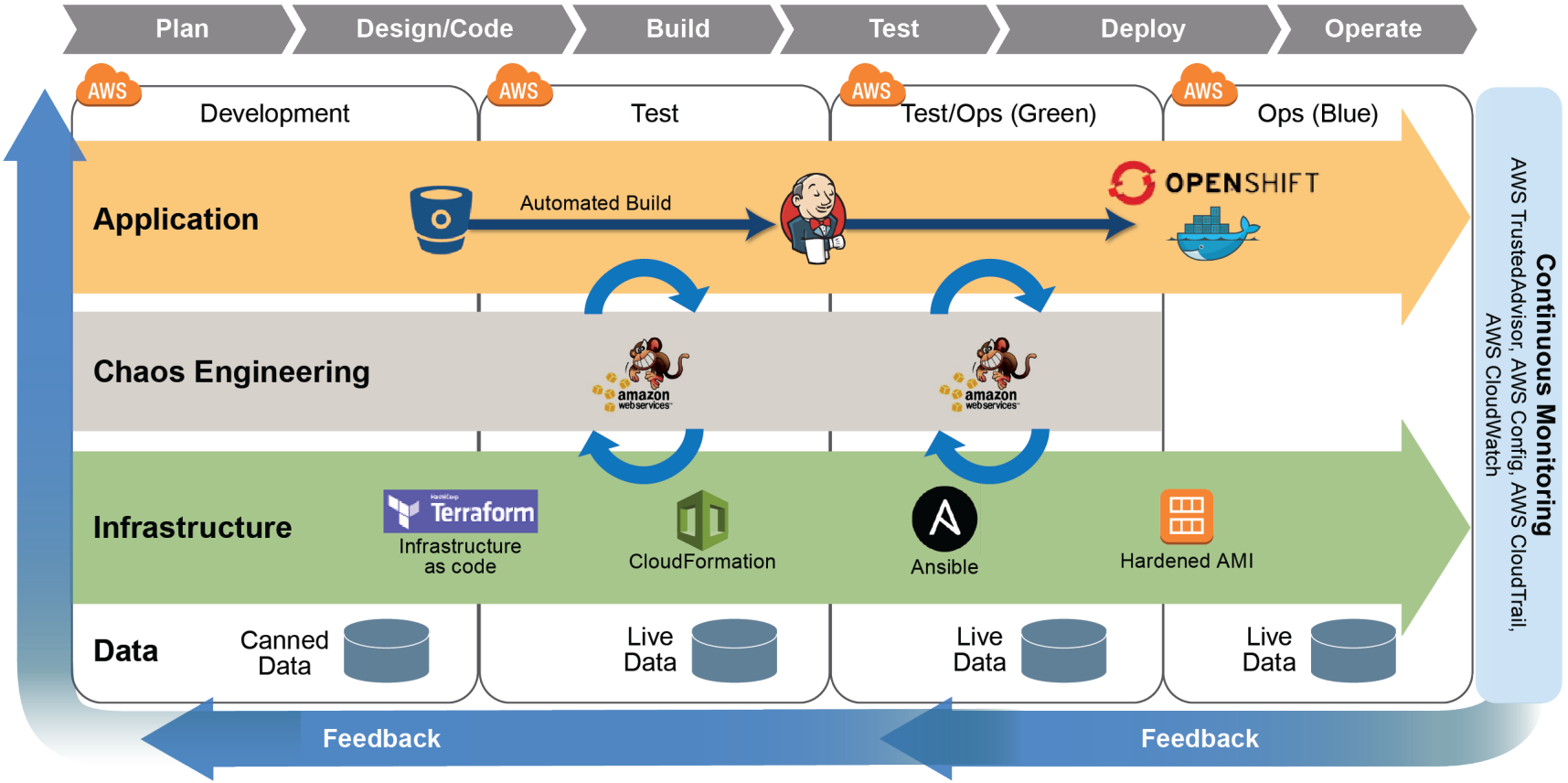
CHAOS MONKEY

Rule	Description
Shutdown Instance	Randomly terminates an EC2 instance
Burn CPU	Sets CPU utilization to 100% for an instance
Block All Network Traffic	Blocks all network traffic to an instance
Kill Processes	Kills all Java & Python processes on an instance
Null Route	Adds IP routing to a black hole
Fail EC2	Makes the EC2 service endpoint unreachable
Fail S3	Makes the S3 service endpoint unreachable
Network Corruption	Corrupts a percentage of all network traffic
Network Latency	Adds a delay to all network traffic
Network Loss	Drops a percentage of all network packets

CHAOS MONKEY

Rule	Description
Burn IO	Causes disk activity on the root disk to spike
Fill Disk	Fills the root disk with junk data
Fail DNS	Drops all port 53 traffic
Fail Dynamo DB	Makes the Dynamo service endpoint unreachable
Detach EBS Volume	Detaches EBS volume from running instances

REFERENCE ARCHITECTURE



CLOUD RESILIENCY BEST PRACTICES

- **Auto Scaling Groups-** Maximize benefits of the cloud by improving fault tolerance, increasing availability and enabling better cost management
- **Block Storage-** Automatically replicate Elastic Block Storage (EBS) to protect from component failure and utilize snapshots
- **Immutable Infrastructure-** Design servers and other components as temporary resources. Servers should be replaced versus updating resulting in resources always being in a consistent state
- **Load Balancing-** Distribute incoming traffic across multiple EC2 instances over multiple availability zones
- **Loose Coupling-** Break applications down into smaller loosely connected components to make more maintainable and less prone to failure
- **Multiple Availability Zones-** Reduce the risk of outages due to loss of an availability zone by enabling high availability across availability zones

CLOUD RESILIENCY BEST PRACTICES

- **Persistent Data-** Enable data persistence through Relational Database Service (RDS) with multiple availability zone deployments and master/slave nodes to enhance data availability and durability
- **Remove Single Points of Failure-** Design highly available systems to function even after the failure of loss of individual or multiple components/ services
- **Security-** Manage user roles/access using IAM and KMS services and configure resources to protect data at rest and in transit
- **Instance Uptime-** Shutdown resources at night or during periods of non usage in order to reduce costs and reduce attack footprint
- **Automation-** Automate instance/ resource provisioning and resizing to meet utilization needs and reduce costs
- **Management Tools-** Leverage CloudWatch, CloudTrail, Config, Trusted Advisor, Cost Allocation Tags

QUESTIONS?