PREPARING FOR DISASTER

INTEGRATING BCDR PRINCIPLES INTO YOUR DEVOPS PRACTICE

Jeremy Heffner

SANS Secure DevOps Summit & Training
October 2017
AWS console breach leads to demise of service with “proven” backup plan

Code Spaces closes shop after attackers destroy Amazon-hosted customer data.

DAN GOODIN - 6/18/2014, 2:12 PM

MAJOR AWS S3 OUTAGE, 2/2017

The Internet grinds to a halt

We have not completely restarted the index subsystem or the placement subsystem in our larger regions for many years. S3 is designed to support the removal or failure of significant capacity with little or no customer impact. We build our systems to rely on the ability to remove and replace capacity as one of our core operational processes. While this is an operation that we have performed hundreds of times, it is not something we do at the scale we needed to do on February 20th. The process of restarting these services and running the necessary safety checks to validate the integrity of the metadata for the first of the two affected subsystems that needed to be restarted. By 12:26 PM PST, the index subsystem had activated enough capacity to begin serving clients...

Source: https://aws.amazon.com/message/41926/
WHAT IS BCDR?

• **BUSINESS CONTINUITY**
  • “CAPABILITY OF THE ORGANIZATION TO CONTINUE DELIVERY OF PRODUCTS OR SERVICES AT ACCEPTABLE PREDEFINED LEVELS FOLLOWING DISRUPTIVE INCIDENT” [SOURCE ISO 22300]

• **DISASTER RECOVERY**
  • **TECHNOLOGY TO ENABLE THE BUSINESS CONTINUITY PLAN**
DISASTER - CAUSES

- NATURAL
  - POWER FAILURES, STORM SYSTEMS, EARTHQUAKES
- TECHNOLOGY
  - BUGS, HARDWARE FAILURE
- MAN-MADE ACCIDENTS
  - BAD CODE, TYPoS, BACK-HOE, FISHING TRAWLER, WRONG ACCOUNT
- MAN-MADE MALICIOUS
  - WAR, HACKERS, DISGRUNTLED EMPLOYEES
DISASTER - EFFECTS

- Data loss
- Systems crash
- Connectivity loss
- Service outage
- Loss of revenue
- Reputation damage
- Worse
TRADITIONAL DISASTER RECOVERY

- Backups
- Offsite Storage
- Failover sites
- Hot/Cold Standby
- Everything here is over there
- Binders full of runbooks
- People with pagers
WHAT ISN’T DR

• **High Availability**
• **Load Balancing**
• **Cloud Backups**
• **Continuous-Integration / Continuous-Deployment**
DR IN THE CLOUD – PAIN POINTS

• OPAQUE CLOUD SERVICES
• DEFINING “OFFSITE” IS DIFFICULT
• VENDOR LOCK-IN
• DATA/Network TRANSIT GETS COST PROHIBITIVE
• RELIANCE ON COMPLEX CI/CD PIPELINES
• CONNECTIVITY
CLOUD DR USING THE CLOUD

• Cloud storage for Backups
  • Store it as a backup, not just on a cloud service
• Off-Site Backups
  • Diversify technology / Provider
  • Authentication domain
  • Diverse Environments
• Automation Systems
• Interfacing different services
PATH TO SUCCESS

• THE #1 RULE FOR DISASTER RECOVERY

TEST IT
NETFLIX – 100% IN THE CLOUD!

• **No datacenter for local backups**
• **Hosted in AWS – location diversity but no tech diversity**
• **Heavily automated deployment and integration path**
• **Constantly learning from failures, experiments**
• **A/B testing is built into everything**
• **Many microservices, no monolithic services**
NETFLIX – RESILIENCY TESTS

• Chaos Monkey
• Chaos Kong
• Armageddon Monkey
NETFLIX – DISASTER RECOVERY EXERCISES

- Based on and parallel to existing ERM, BCP, DR plans
- Team of engineers from key areas
- Separate AWS environment
- Start up basic services
- Moving target
- Document dependencies, procedures
NETFLIX – ARMAGEDDON PROJECT GOALS

- **Automate Runbooks with Python**
- **Validate Service Assumptions**
- **Programmatically Verify Dependency Graphs**
- **Experiment to Learn Resiliency**
- **Determine What Data We Need to Backup**
ARMAGEDDON PROJECT – FIRST HURDLES

- Update Runbooks
- Backup configuration and environmental Data
- Backup CI/CD pipeline configurations
- Identify bootstrap data
- Build foundational services and tools
ARMAGEDDON PROJECT – BACKUP SERVICE

• Stores important data offsite
• Technology / provider diversity
• Integrated with CI/CD tools
• Object level encryption and validation
• Retains history
• Server-less AWS primitives
ARMAGEDDON PROJECT – ENV/CONFIG BACKUP

• Utilizes the Armageddon backup service
• Gathers data via SecOps tooling
• Gathers CI/CD DevOps pipeline configuration
ARMAGEDDON – ENV/CONFIG REPLICATOR

- **Utilizes** configuration backup data
- **Configures** fresh AWS environment
- **Whitelisted** set of services
- **Configures** needed network, security group, and roles
- **Tears down** environment after exercise is completed
ARMAGEDDON PROJECT – AUTOMATED EXERCISE

- **Python script with defined dependency graph**
- **Brings services on-line in proper order**
- **Scripted post configuration**
- **Verifies services are running, starts next services**
- **Provides alerting on failure**
- **Designed to be run in a loop via AWS Lambda**
- **Reporting back via SNS to a Dashboard (WIP)**
ARMAGEDDON PROJECT – SUCCESS

• **Clean AWS account in non-production region**
• **Automated Backups and Restores**
• **Initial set of core Netflix services automated and tested**
• **Ability to detect and identify failures in service automation**
• **Initial ties into main DevOps and SecOps pipelines**
EXPERIMENT FINDINGS – THE GOOD

- **Existing DevOps pipeline helps to find the microservices**
- **Having strong APIs for CI/CD tools enables DR automation**
- **Microservices help with fault isolation and identification**
- **Scaling down for a test is easy**
- **A DR DevOps pipeline can keep up with developer velocity**
EXPERIMENT FINDINGS – THE BAD

• MANY MICROSERVICES MEAN MANY DEPENDENCIES
• MAPPING FULL THE DEPENDENCY GRAPH IS PROBLEMATIC PAST 1 DEEP
• A/B TESTING MEANS THE SERVICES IS NEVER OFFLINE. EVER.
• FAULT ISOLATION FAILS WITH COMMON LIBRARY AND APPLICATION SERVERS
• DATA CREATED IN THE CLOUD IS HARD TO MOVE OUT OF THE CLOUD
• REGRESSIONS ARE COMMON
HARDENING MICROSERVICES

- Automated isolated testing environment
- Extend Chaos like tests to service tiers
- Increase traceability of services to analyze dependencies
- Static analysis of code and build systems for dependencies
- Automate any configuration or migration tool
- Discourage developers from system access
Disaster Recovery in DevOps

- Prove your backups work – Every day!
- Extend A/B testing ideas to DR testing
- DR systems provide feedback to the CI/CD pipeline
- Isolate and test tiers/grouping of services
- Encourage service providers with APIs and export mechanisms
QUESTIONS?
THANKS!

JEREMY.HEFFNER@VIASAT.COM