CSV-W14 - BUILDING AND ADOPTING A CLOUD-NATIVE SECURITY PROGRAM

Rich Mogull
VP of Product/Analyst
@rmogull

Bill Burns
Chief Trust Officer, VP Business Transformation
Informatica
@x509v3
Our Plan

- Show you how to build your program from scratch
  - And bridge existing capabilities

- Highlight fundamental differences of cloud

- Show you what’s real
  - We’ve done this… all of it. None of this is theory

- From architecture and strategy to key controls areas
STRATEGY
Cloud Native == NOT.Legacy

You build it from scratch  OR  Redesign without compromise
Building the cloud-native mindset

Cloud is:
- Developer Driven
- Abstracted
- Automated
- Volatile
- Ephemeral
- Massively Scalable

You should:
- Embrace Automation
- Embrace Failure
- Think Like a Developer
- Kill the Past
- Isolate by Default
Know the Lines, Use Them for Advantage

- Your customers and competitors are already getting comfortable with this

Risk & Compliance
- Declare and understand the shared responsibilities model
  - What you are responsible for, what the provider is responsible for.
- Critical cloud provider security capabilities
  - You interact via APIs, not projects
- *Use the model to change attack surface*
Critical cloud provider security capabilities

- API/admin activity logging
- Elasticity and autoscaling
- APIs for all security features
- Granular entitlements
- Good SAML support
- Multiple accounts/subscriptions/projects per customer
- Software defined networking
- Region/location control
Cloud Native Security Program Principles

- APIs
- Automation
- Immutability & Isolation
APPLICATION SECURITY

App security before infrastructure
Cloud Application Security Process

- Secure architecture and design
- Secure deployment/DevOps
- External security controls
- Secure Operations
Even infrastructure security starts with application design

- Fit the infrastructure to the application
- Leverage architecture for security
- Automate deployment and management with DevOps
  - Required for immutable, cloud disaster recovery, and portability anyway
- Good design can eliminate common traditional security issues
- Network attack paths, patch management
Secure Application Development/Deployment (Example)

- Source Code
- Cloudformation Templates
- Chef Recipes
- Git
  - Functional Tests
  - NonFunctional Tests
  - Security Tests
  - Jenkins
  - Test
    - Unit/regression tests
    - Functional tests
    - SAST
    - DAST
    - VA/Network/Infrastructure
  - Prod
Architecture + DevOps is cloud security

- Architecture allows you to leverage the shared responsibilities model to reduce your *security management surface* and push them onto a cloud provider that has stronger economic incentives to avoid incidents.
  - You are paying for it anyway, might as well take advantage.
- DevOps provides a **consistency** and **control** impossible with manual application deployments.
- Security can easily **embed** and **automate** into the same toolchains as development and operations.
- Security can steal DevOps techniques to improve security operations.
INFRASTRUCTURE

Building your cloud infrastructure security
Start with Architecture

- Architecture == security; it’s a primary control not something to eval after the fact
- Intrinsically tied to applications: These are general principles and cross-project design patterns.
- Segregate with Accounts/Subscriptions/Projects first, virtual networks second
- Minimize blast radius
- Leverage immutable infrastructure
- Cut off network attack paths
- Wipe out classes of traditional security issues
Own the Management Plane, Own the Cloud

- Baseline shared services / management plane
- Use to enforce guardrails - let the cloud move fast but detect and respond quickly to things that stray too far
- Make this “more secure approach” the “easier approach” to adopt. Enforce as the pathway in / out.
Define infrastructure in code

Containers

Servers (Instances)

Full Stack (Networks/IAM/etc)
Infrastructure As Code - Master Image Bakery

- InfraSec Requirements
- Packer configuration
- Git
- Jenkins
- Security Tests
- Test
- Master Image

Replace, Don’t Patch
Immutable Infrastructure

Template B:
Baseline Shared Services

- Federated Identity Broker
- ABAC
- Security Access
- Multiple authorities

Identity Provider

- Federated Trust (e.g. SAML)
- Directory server holding identities
  - Authenticates users
  - Provides required attributes
- Maps identities and attributes to internal model
- Authorization
- Session management

IAM

Relying Party

AuthN

AuthZ
Baseline Shared Services

- Centralize
  - But allow local access
- Leverage cloud events for real-time alerts
  - Use to trigger guardrails
- Cascade and filter to manage costs
Example: Data Analytics Pattern

Attack surface?

Network attack paths?
Cloud Network Security

- Insulate with load balancers, ASG, PaaS
- Architect with native security group and route patterns
- Supplement gaps
- Automate + guardrails

Fit to the app
Cloud Workload Security

Function as a Service (Lambda)  Instances/VMs  Containers

Security Focus:  Code and config  Trad or Immutable  Code + Hardening
Workload Security Wins

- Availability - embrace chaos engineering & immutability
  - Abolish fragile points in the architecture; design auto-scale groups for all critical services
  - Immutable and micro services
  - Serverless and FaaS
### Instance (and container) security issues

#### Vulnerability and Patch Management
- Limitations on scanning due to provider terms of service
  - Or use a cloud-compatible host agent
- Can scan in deployment pipelines with immutable
- Patching on running instances only really used with standard/traditional instances
  - Weakest security option
- Can't rely on scans reflecting the actual system if they are IP based

#### Logs and Security Tools
- Logs can't rely on hostnames or IP addresses
  - More on logging later, but you need to build a cloud-native architecture
- Security tools also can't rely on normal addresses due to the high rate of change in cloud
  - Also they need to be lightweight since processors == cash
  - Ideally deployable with CI/CD
  - Need to auto-self-configure and connect to management systems
- EPP/AV often not needed

#### Other
- Images can expose data if they aren’t created properly
  - This is really easy to mess up
- Snapshots are ideal for data exfiltration
  - Encryption knocks down this problem
  - Encryption options generally built into the cloud platform, don’t need external tools
- Instances can access metadata on themselves which can be used for attacks if not properly managed
Encryption is the default. Focus on key management.
Cloud encryption

- Encryption is easy … key management is Hard
- Do NOT do this yourself or roll your own
- Focus on **BYOK** for integrated encryption
- Cloud HSMs and KSMs - your cloud keys can be more secure than in most data centers
For regulated data, app-level encryption is your friend.

Integrate with IAM and automation.

Know the lines where data is potentially exposed and if it matters.

Cloud HSMs and KSMs - your cloud keys can be more secure than in most data centers.

Option A: KMS encrypts the data.
Option B: App encrypts with KMS key.
Option C: KMS integrated with CSP Storage.
SECURITY OPERATIONS

Monitoring/alerting, incident response, assessment, management
Alerting tips

- If your cloud provider offers a security threat service… use it!
  - They see things you can’t, such as failed API calls
- In-cloud event-based alerts are fastest
  - But also need to be configured and managed in each cloud deployment
  - Use automation to build/support
- Feed alerts to central feed, or alert off a central feed
  - We discussed earlier
Alerting++

Change a security group

Event Recorded to CloudTrail

Passed to CloudWatch Log Stream

Triggers an CloudWatch Event

Lambda Function analyzes and reverses

AND/OR

Notification
Assessment

Cloud Configuration

• Some providers have built-in tools
• Third-party
• Manual assessments for context

Instances/Containers

• With immutable assess in the pipeline
• Prefer agent over network based

Application

• Push most testing to CI/CD
• Full web testing may require permission
• Choose pen testers with cloud experience
Incident Response

- Cloud registry
- Who to call and out of band authentication
- Cloud IR jump kit centralized automation is ESSENTIAL!
- Attack and response are all over API
- Focus on the management plane first
- Lock down IAM before anything else
- Multiple accounts dramatically improve IR success rates
- Did I mention automation? Quarantine environments?
Automate security management

- Infrastructure templating should include security
  - This is how you handle new account provisioning

- Use the hierarchical controls of your cloud provider
  - Set a security role to gain account access

- Use automation for cross-account and multi-cloud management
  - Guardrails
  - Workflows
  - Orchestrations (integrating 3rd party tools to operations)
Automate security management

- Collections of scripts/code, not a monolithic application.
- Framework for:
  - Auth
  - Scheduled activities
  - Logging
Adopting and adapting to cloud security

- **First 30 days**
  - Find your cloud controller - Take owner to lunch; pain points
  - Get a read-only account. Get comfortable with APIs for cloud inventory, drift detection, attack surface changes, image configs.

- **Within 2 months**
  - Create a master image bakery
  - Get good at pushing new images on demand, work with app teams to rely on your images for push

- **Within 3 months**
  - Create guardrails to detect and alert on cloud config drift/violations
  - Automate remediation for simple failure cases, more complex checks over time