DevSecOps: Getting There From Here

Dave Shackleford
SANS and Voodoo Security
Meet Kyle

• Kyle works in information security
• Like MOST infosec professionals, Kyle is:
  – Witty
  – Handsome
  – Brilliant
However...

- Kyle has a PROBLEM.
- Kyle’s organization is moving to CLOUD.
- Kyle has to figure out CLOUD SECURITY.
Dev and Ops and DevOps

- KyleCo has a Dev team.
- KyleCo has an Ops team.
- The CIO is a fan of this new concept called “DevOps”.
- Kyle has no idea what he’s talking about.
- But Kyle can learn.
Kyle’s Journey

• DevOps is all about:
  – Automated … everything.
  – Infrastructure-as-code
  – Orchestration
  – Built-in security services where possible
  – New tools.
  – New pipelines.
  – Less tolerance for “slow security”.

• Kyle learns all this, and more.
So...the problem?

• Kyle learns a few hard truths QUICKLY
  – DevOps works differently
  – DevOps works faster in many ways
  – DevOps expects everything to be integrated

• There’s also a problem with POLITICS
  – The CIO wants to embrace DevOps
  – Politically, DevOps > Infosec
  – Poor infosec.
So Kyle Has to Change His Mojo

• In infosec, we’ve got some big changes to make
• First, we’ve got to build a new governance model in IT and development
• Second, we may need some new skills!
• Finally, we’ll need to understand “security as code” much better
STEP 1: GOVERNANCE
Governance Models

• There are sophisticated governance models out there, but here’s the basics:
  – Monarchy Model (Top Execs, Top-Down)
  – Business Unit Execs / Divisional Model
  – Collaborative / Group Decision Model

• How does cloud and DevOps change these?
What’s Involved in Governance?

• There are many aspects of governance to consider
  – Security policy
  – Procurement and risk assessment practices
  – Change management

• There are many more, but these are the big ones for cloud and DevOps
Security Policy for DevOps

• Do you need a specific policy for DevOps alone?
  – Not likely.

• What goes into a policy?
  – Acceptable cloud deployment models/practices
  – Data types that can/cannot go to the cloud
  – Key/credential/data protection requirements
  – Compliance requirements
Procurement / Risk Assessment

• DevOps and Risk Assessment?
  – Nah, not really

• HOWEVER:
  – Vendor mgmt. and procurement should consider DevOps requirements
  – This can include:
    • APIs
    • Logging/monitoring
    • Support for specific tools/automation models
Changes for Change Control

• Traditional change control really falls apart in a DevOps model

• Security professionals will need to work with DevOps teams to determine:
  – Which changes need traditional tracking
  – Which changes can be “log and review”
  – Review cycles and review board groups
STEP 2: SKILLS
Security Pros Need New Skills

• In its 2016 “State of Cloud Security” report, the Cloud Security Alliance (CSA) acknowledged a significant skills gap in cloud security.

• DevOps integration may require some new skills focus:
  – Coding
  – Orchestration/automation
  – Web services
  – Virtualization/containers
Skills: Coding

• Security pros don’t need to be full-fledged developers
• Some code skills may be helpful, though:
  – Python
  – Ruby
  – Shell scripting
• Being comfortable with formats like YAML and JSON is also important
Skills: Virtualization/Containers

• Most DevOps environments and cloud deployments rely on VMs and containers
• Security teams need to understand how to secure VMs and container technologies
• Key concepts include:
  – Hypervisor lockdown
  – Virtual and software-defined networking
  – Container lockdown and scanning
Skills: Automation and Orchestration

• DevOps teams make extensive use of automation and orchestration tools:
  – Ansible
  – Chef/Puppet/Salt
  – Jenkins
  – Kubernetes/Docker Swarm

• Security teams need to learn:
  – Secrets management
  – Hardening these systems
  – Privilege control
Skills: Web Services/Microservices

- Many security teams are not familiar with microservices
- Microservices are “decoupled” application architectures, often found in DevOps and cloud deployment scenarios
- What’s often included?
  - Containers (sometimes virtual machines)
  - APIs (often RESTful and lightweight)
  - Scalable Cloud Infrastructure (software-defined environments)
STEP 3: SECURITY AS CODE
Security as Code?

• With DevOps and “Infrastructure as Code”, we define everything in a software-defined method:
  – Servers (usually VMs)
  – Containers
  – Application stacks
  – Networks
  – Roles/Privileges/Access models

• Security needs to be defined in this way, as well
Deployment Pipeline Security

• Security teams should focus on:
  – Code security
  – Code repositories’ security
  – Automation tools
  – Orchestration platforms
  – Gateways and network connectivity

• Authentication/Authorization and privileged user monitoring and management are critical
Development/Deployment Integration

• We need to integrate into deployment pipelines
• Continuous Integration vs. Continuous Deployment
• Early: Static and Dynamic code analysis
• Early: Defined libraries and configs
• Later: Monitoring and Control in instances
Security as Code: Define Policies

• Define policies for components, networks, and more

• This might include:
  – Configurations (Puppet, Chef)
  – App deployment and automation (Ansible, Jenkins)
  – Additional orchestration and automation tools

• Cloud providers may offer tools, too (CloudFormation in AWS, for example)
Security as Code: Define security “stories”

• These will be specific use cases and requirements:
  – Input validation for app X
  – Use of TLS for all communications
  – Hardening to CIS Benchmark standards

• These are then implemented IN code and vetted, or via policy files and language
Security as Code: Internal Build and Deployment Security

• For the internal side of Security as Code, imagine the following:
  – Automated code scans upon check-in
  – Automated app scanning in test/staging
  – Automated Server, Container, and Network configuration checks via policy
  – Continuous monitoring of all core components in the Deployment Pipeline
Security as Code: Test policies regularly

- Using build testing tools like Test Kitchen and Vagrant can simplify internal policy validation
- Coordinate penetration tests and routine checks to validate policies’ effectiveness
  - Are only required ports open?
  - Are credentials secured?
  - Are encryption keys secured?
  - Are privileges assigned properly?
Security as Code: Automate Production Feedback Loops

• Proper DevSecOps needs continuous monitoring

• You need detection and response playbooks, too:
  – Scheduled checks of X generates alert/log
  – Alert triggers automated process Y

• All of this needs to be automated
  – Some critical tasks may require a human sign-off
Wrapping Up

• So – how do we get to DevSecOps?
• First, adjust your governance model
• Second, invest in new skills
• Third, embrace “security as code” to integrate into dev and deployment pipelines
• DevOps teams need security integrated into their processes – it’s our job to get this done!