DevSecOps: An organizational primer

Tim Anderson
Sr. Technical Industry Specialist,
AWS Security
Outcomes

1. The business case for DevSecOps
2. Design of your foundational DevSecOps team
3. Security mechanisms for scaling DevSecOps success
What Is Culture?

Culture is “the Way We Do Things Around Here.”

• Culture is the “software of the mind.” It is the core logic that organizes people’s behavior.

• The culture reflects the lessons learned that are important enough to pass on to the next generation.

• Values, beliefs, and practices that have been developed and reinforced over time.
Competing Forces

Business

Development
- Build it faster

Security
- Make it secure

Operations
- Keep it stable
90% of CEOs believe their industry will be digitally disrupted

Less than 15% are well-positioned to execute on a digital strategy*

*McKinsey
70% of customer challenges are non-technical.
Building the business case for DevSecOps transformation
What is DevSecOps?

DevSecOps is the combination of cultural philosophies, practices, and tools that exploits the advances made in IT automation to achieve a state of production immutability, frequent delivery of business value, and automated enforcement of security policy.

DevSecOps is achieved by integrating and automating the enforcement of preventive, detective, and responsive security controls into the pipeline.
Tenets of DevSecOps

1. Everyone is a security owner

2. Test security as early as possible to accelerate feedback.

3. Prioritize preventive security controls to stop bad things from happening.

4. When deploying a detective security control, ensure it has a complementary responsive security control to do something about it.

5. Automate, automate, automate.
The Benefits

- Fast time to market or time to value
- Less waste from producing capabilities
- Less waste in processes
- Reduced risk
- Increased innovation
- Better operational controls through automation
## Traditional IT vs. Full Stack Engineering

<table>
<thead>
<tr>
<th>Challenges</th>
<th>Traditional IT activity-based teams</th>
<th>Full-Stack Engineering outcome-based teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handoffs</td>
<td>Application Development</td>
<td>Application Development</td>
</tr>
<tr>
<td>Resource Constraints</td>
<td>Application QA &amp; Testing</td>
<td>Application QA &amp; Testing</td>
</tr>
<tr>
<td>Competing Priorities</td>
<td>Middleware &amp; Integration</td>
<td>Middleware &amp; Integration</td>
</tr>
<tr>
<td>Wait Time</td>
<td>Database</td>
<td>Database</td>
</tr>
<tr>
<td></td>
<td>Network</td>
<td>Network</td>
</tr>
<tr>
<td></td>
<td>Storage</td>
<td>Storage</td>
</tr>
<tr>
<td></td>
<td>Compute</td>
<td>Compute</td>
</tr>
</tbody>
</table>

### New Burdens

- Infrastructure
- Operations
- Security
- Finance

© 2019, Amazon Web Services, Inc. or its Affiliates.
Innovation drain

In the process, pervasive handoffs, bottlenecks, and defects are created.

Defects passed downstream are often discovered late in the delivery cycle and have to be revisited.

Each Step Delays Time to Value
How to Influence Cultural Change?

- Identify desired attitudes and behaviors for successful cloud adoption
- Communicate attitudes and behaviors
- Align explicit and implicit reward systems
- Align hiring, training, and incentive practices
# Driving Change - Area of Focus

<table>
<thead>
<tr>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIPPO-based decision-making</td>
<td>Data-driven decisions that are tested and measured</td>
</tr>
<tr>
<td>Large feature sets and systems sprawl</td>
<td>Constantly re-prioritizing and validating for relevance</td>
</tr>
<tr>
<td>Protecting the core business</td>
<td>Continuous refactoring and improvement</td>
</tr>
<tr>
<td>Business and IT silos</td>
<td>Teams that span business and technology</td>
</tr>
<tr>
<td>Big bets that languish</td>
<td>Reduced batch size and frequency of releases</td>
</tr>
<tr>
<td>Software and processes that aren’t nimble</td>
<td>Reducing the lead time from idea to implementation</td>
</tr>
<tr>
<td>Planning for best case operating state</td>
<td>Assuming attack and failure</td>
</tr>
<tr>
<td>Gated opaque security slows the business</td>
<td>Security as quality - business driver and differentiator</td>
</tr>
</tbody>
</table>
Be aware of top 5 pitfalls

1. Lack of Executive Sponsorship
2. Poor Communications
3. Insufficient Resource Allocation
4. Undefined KPI’s and Outcomes
5. Timing
Building your foundational DevSecOps team - a security focused CCOE
Commission a CCOE

A two-pizza, empowered, and accountable team that owns the cloud strategy, establishing the cloud service, and helping the business / dev teams migrate their first few applications

- Cross-functional / Hands-on
- Product focused
- Dedicated
- Empowered

- A change agent
- Creates roadmap
- Establishes standards
- Partners with early-adopters
Driving Change with The CCOE

- Building reusable patterns / Product focused
- Ingraining security with every team member
- Visibility of team operations
- Continuous improvement – feedback cycle and actions
- Look to simplify
Structure of Product Teams

How do we structure these product teams?
How do you start your CCOE?

Think big, but start small. Launch a cloud foundation team and a small number of development teams to start the flywheel. Scale as the customer’s cloud transformation accelerates and expands.
What products does cloud platform engineering provide?

Cloud platform engineering (CPE) products

<table>
<thead>
<tr>
<th>Codified patterns</th>
<th>Core platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise “stacks”</td>
<td>CaaS/FaaS</td>
</tr>
<tr>
<td>Configuration management</td>
<td>Core networking</td>
</tr>
<tr>
<td>Primitives</td>
<td>Accounts, IAM &amp; SSO</td>
</tr>
</tbody>
</table>

Build, test & deploy

<table>
<thead>
<tr>
<th>CI/CD &amp; release management</th>
<th>Telemetry, alerts &amp; insights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration management</td>
<td>Patch, backup &amp; restore</td>
</tr>
<tr>
<td>Source code &amp; artifact repositories</td>
<td>ITSM &amp; self-service</td>
</tr>
</tbody>
</table>

Define & enforce

<table>
<thead>
<tr>
<th>IAM &amp; policy management</th>
<th>Threat &amp; vulnerability management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network security</td>
<td>Security information &amp; event management</td>
</tr>
<tr>
<td>Secrets &amp; encryption</td>
<td>Incident response &amp; forensics</td>
</tr>
</tbody>
</table>

Operate & manage

<table>
<thead>
<tr>
<th>Telemetry, alerts &amp; insights</th>
<th>Patch, backup &amp; restore</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITSM &amp; self-service</td>
<td>ITSM &amp; self-service</td>
</tr>
</tbody>
</table>

Detect & respond

| Threat & vulnerability management                       |                                     |
|--------------------------------------------------------|                                     |
| Security information & event management                |                                     |
| Incident response & forensics                          |                                     |

Cloud platform engineering codifies differences between stock AWS service configurations and the enterprise’s standards, packaged and continuously improved as self-service deployable products to customers.
How do you re-envision the world as products?

Two kinds of products:

Experiences

- Adaptive Home Page

Services

- Search
- Cart
- Account
- Item
- Advertising
- Promotions
- Digital Asset
- Others...

© 2019, Amazon Web Services, Inc. or its Affiliates.
Products are delivered by stable “product teams”
A deeper look at enabling people

- Everybody is a Security Engineer
- Pair Programming Works
- Tooled Correctly for Continually Learning
- Certification Rules
- Recruit for Alignment with your Tenets
- Recognize what Motivates Engineers and Developers
Security mechanisms for DevSecOps
Security Blind Spots

- Disparate sources
- Lack of rigor
- Can't scale

© 2019, Amazon Web Services, Inc. or its Affiliates.
Get Humans Away from Your Data
Instead of “No”, security should say “how can we do this?”

- Change from gating to guardrails
- Establish norms for security hygiene and set high quality standards
- Craft policy enabling teams to operate freely within the determined constraints – stepping towards continuous authorization.
- Consistently communicate the connection between security and business objectives
Giving security confidence – Proving Assurance

- Threat modeling
- Feed security cases to the Dev team - work it like high priority defects
- Address separation of duties concerns
- Adopting zero known defect approach
- Continuously vet/audit security in dev and prod
  - Rigorous testing in each environment
  - Peer review - Each technologist should be thinking about possible defects and possible security vulnerabilities. Code should always be reviewed by a peer, who should also be looking for vulnerabilities
Source
- Version Control
- Branching
- Code Review

Build
- Compilation
- Unit Tests
- Static Analysis
- Packaging

Test
- Integration Tests
- Load Tests
- Security Tests
- Acceptance Tests

Production
- Deployment
- Monitoring
- Measuring
- Validation
Team Interaction and Workflow

1. On-Boarding, Coaching, & Product Feedback
   - Populate Revise Prioritize Backlog
   - Lifecycle Management of Application Stacks

2. Pull Approved Platform Artifacts from Shared Repositories
   - Extend Approved Artifacts for App Stacks
   - Security Focused Code Review

3. Operate Application Stacks
   - Operate Application Stacks
   - Build, Test, and Deploy App Stacks

4. Integrate & Deploy Cloud Services
   - Automated Security Testing

5. Operate Cloud Services
   - Service Catalog or Shared Repositories
   - Secure/Hardened Environments

6. Publish Artifacts & Documentation for Cloud Services
   - Threat Modelling
   - Automated Security Testing

Continuous Delivery of Cloud Services
- Build & Test Cloud Services
- Integrate & Deploy Cloud Services

Continuous Delivery of Application Services
- Security Focused Code Review
What does it look like?

1. Scan for creds
2. Static analysis
3. Logic / Library scan
4. Smoke test
5. Deploy into repo
General best practices used by Amazon developers

- CI/CD is a MUST!
- Clean room
- Everything that is code goes into a repository
- Start with continuous delivery
- Deploy to canaries, test, deploy to an AZ, test, deploy to a Region, test
- Code Reviews are one of the best mechanisms for “good” code
- Style checkers
- Auto-rollbacks can be the quickest recovery mechanism after failure
- Thorough dashboards
Consistency Breeds Trust

- Normalize processes and tech stack
- Release management
- Configuration management
- normal vs. abnormal behavior
- Maintain disciplined ITSM use

CI/CD
Using CI/CD to Drive Cultural Security Milestones

- Deeply understand your SDLC
- Catalog the controls
- Document every instance of human interaction
- Reduce human access
- Set a goal to deploy workloads from source.
Let (Security) Builders... Build!
Thank You

Tim Anderson

tdander@amazon.com