How leaders set the stage - successfully scaling DevSecOps

Tim Anderson
Sr. Security Advisor,
AWS Security
tdander@amazon.com
Outcomes

1. A strategy for scaling adoption
2. Mechanisms to build security at scale
DevSecOps strategy that scales
Defects passed downstream are often discovered late in the delivery cycle and have to be revisited.

In the process, pervasive handoffs, bottlenecks, and defects are created.
Traditional IT vs. Full Stack Engineering

**Traditional IT**

- Activity-based teams
- Application Development
- Application QA & Testing
- Application Runtime
- Middleware & Integration
- Database
- Network
- Storage
- Compute

**Challenges**
- Handoffs
- Resource Constraints
- Competing Priorities
- Wait Time

**Full-Stack Engineering**

- Outcome-based teams
- Application Development
- Application QA & Testing
- Application Runtime
- Middleware & Integration
- Database
- Network
- Storage
- Compute

**Full-Stack Engineering Team**

**Considerations**
- Infrastructure
- Operations
- Security
- Finance
The Benefits

- Fast time to market or time to value
- Lower costs
- Less waste in processes
- Reduced risk
- Increased innovation
- Better operational controls through automation
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.
Driving Change - Area of Focus

FROM

HiPPO-based decision-making  
Large feature sets and systems sprawl  
Protecting the core business

TO

Data-driven decisions that are tested and measured  
Constantly re-prioritizing and validating for relevance  
Continuous refactoring and improvement

Business and IT silos  
Big bets that languish  
Software and processes that aren’t nimble  
Planning for best case operating state  
Gated opaque security slows the business

Teams that span business and technology  
Reduced batch size and frequency of releases  
Reducing the lead time from idea to implementation  
Assuming attack and failure  
Security as quality - business driver and differentiator
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. Workforce Management
How the team drives change

- Building reusable patterns / Product focused
- Ingraining security with every team member
- Visibility of team operations
- Continuous improvement – feedback cycle and actions
- Look to simplify
Think big, but start small. **Launch a cloud foundation team** and a small number of development teams to start the flywheel. Scale as the 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>

<table>
<thead>
<tr>
<th>Build, test &amp; deploy</th>
<th>Operate &amp; manage</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI/CD &amp; release management</td>
<td>Telemetry, alerts &amp; insights</td>
</tr>
<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>

<table>
<thead>
<tr>
<th>Define &amp; enforce</th>
<th>Detect &amp; respond</th>
</tr>
</thead>
<tbody>
<tr>
<td>IAM &amp; policy management</td>
<td>Threat &amp; vulnerability management</td>
</tr>
<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>

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.
## Critical Success Factors for Successful Transformation

<table>
<thead>
<tr>
<th>Visible and committed leadership</th>
<th>Targeted and effective communications</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;management driving the change&quot;</td>
<td>&quot;adapting the communication strategy&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compelling need for change</th>
<th>Single program focus</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;establishing a high enough sense of urgency&quot;</td>
<td>&quot;prioritizing projects and allocating resources&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clarity of direction</th>
<th>Measurable goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;grounding the vision of the desired state&quot;</td>
<td>&quot;setting reachable milestones&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Broad-based participation</th>
<th>Disciplined project management</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;engage key impacted audiences&quot;</td>
<td>&quot;running the project effectively&quot;</td>
</tr>
</tbody>
</table>
Security mechanisms for DevSecOps
Organizational change

- Move Security up the value chain
- Security as quality
- Lead communities of practice
- Ensure cloud awareness
- Not a team of "no"
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
General best practices

CI/CD is a MUST!

Clean room

Everything into a repository

Start with continuous delivery

Deploy to staging, 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-rollback

Meaningful dashboards
Consistency Breeds Trust

CI/CD

• Deeply understand your SDLC
• Catalog the controls
• Document every instance of human interaction
• Reduce human access
• Set a goal to deploy workloads from source.
Define and Govern the Policy
Interpret Regulation
Define Control Objectives: “What”
Review Control Effectiveness
Respond at Scale
Report Aggregate Risks

Interpret Control Objectives: “How”
Implement Controls
Provide Visibility into Control Status
Monitor Controls
Respond to Control Failure
Thank You!

Tim Anderson
Sr. Security Advisor,
AWS Security
tdander@amazon.com