Optimizing Capability Maturity for Application Security in the Software Development Lifecycle

Jonathan Davis, CISA, CISSP, CCSK – *Vice President, Marketing*

Steve Wolf - *Vice President, Application Security*

AsTech Consulting www.astechconsulting.com

Governance, Risk & Compliance – G22

ISACA®
Trust in, and value from, information systems
San Francisco Chapter

Course Objectives

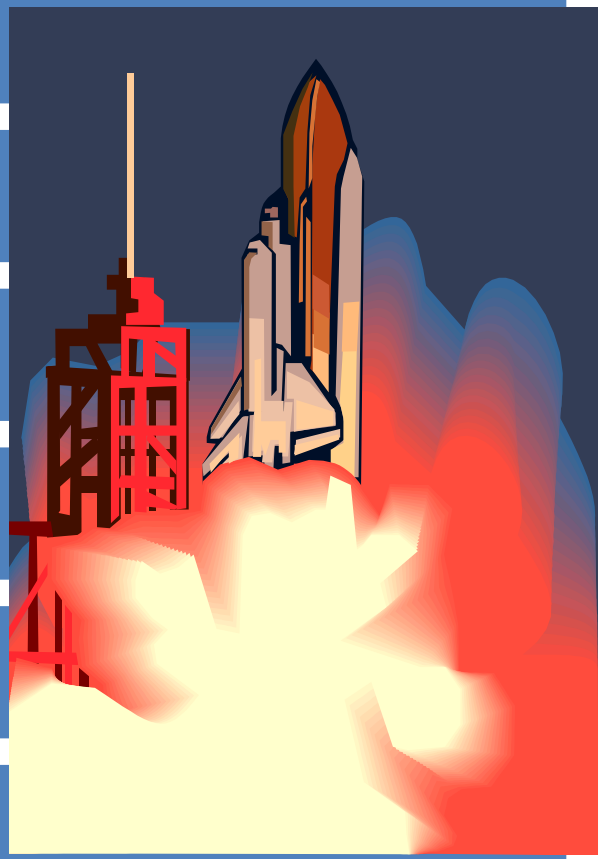
- Intro to Software Security Capability Maturity
- Benefits of Capability Maturity
- Guiding Concepts
- Key Areas of Focus

INTRODUCTION TO SOFTWARE SECURITY CAPABILITY MATURITY

ISACA®
Trust in, and value from, information systems
San Francisco Chapter

Web & Mobile Applications: High Reward ***and*** High Risk

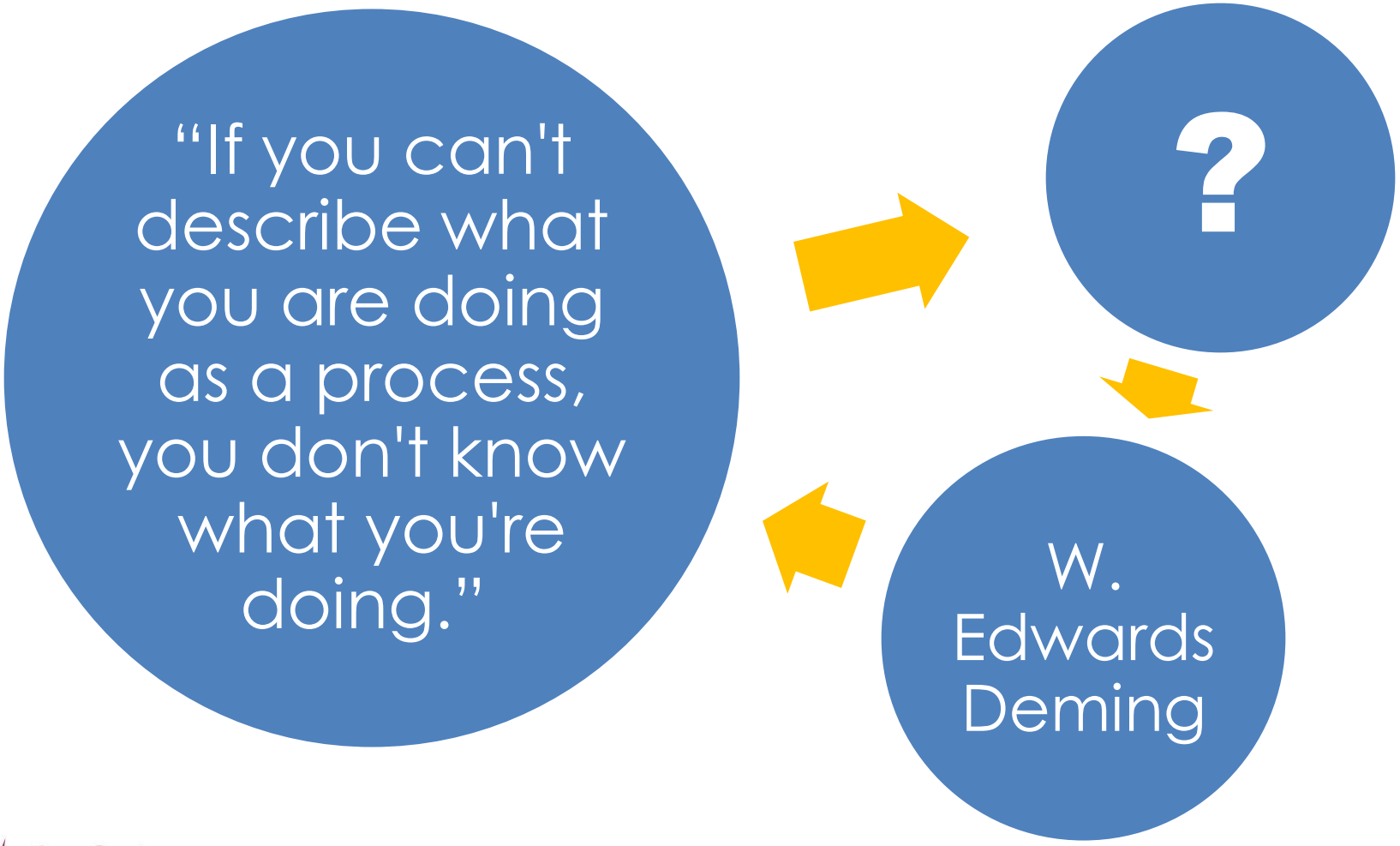
- **Web & Mobile Apps drive revenue**
- Time-to-Market & UX vs. Security
- Attacks & breaches on the rise
- Compliance increasingly complex
- Vendor / 3rd party code risks
- Post-release remediation costly



What Are You Willing to Risk?

- Brand Reputation?
- Loss of Revenue & Customers?
- Fines for Non-Compliance?
- Litigation Exposure?
- High Costs and Post-Release Fixes?

How Strong is Your Faith In Your Application Security Program?



Status Quo is Unacceptable

In an immature organization, there is no objective basis for judging product quality or for solving product or process problems. Therefore, product quality is difficult to predict.

- Software Engineering Institute

Capability Maturity Model for Software

Security is More Than Tools

- A well-crafted **process** is essential
- Depends upon clearly defined **goals**
- **Communication** between stakeholders
- **Metrics** for all key program aspects
- Process must be designed to **evolve**

Enter Capability Maturity

Capability Maturity Models (**CMM**) help establish relevant, repeatable processes that lead to measurable, predictable outcomes. These generate feedback that can be monitored, interpreted, and ultimately optimized for greater quality and efficiency, leading to a virtuous cycle of ***continuous improvement***.

Capability Maturity Models

- Original CMM created in the 1980's
- Joint project of Dept. of Defense and Software Engineering Institute (SEI) at Carnegie Mellon
- Established standardized quality best-practices for software development initiatives
- Software Security CMMs take similar approach, but focus on security in the SDLC

Two Software Security CMMs

- BSIMM
- OpenSAMM

While there are other maturity models for secure development frameworks, such as the **Microsoft Security Development Lifecycle** and **CLASP**, the two models above offer a very practical mix of strategy and detail.

BSIMM

Building Security In Maturity Model

http://www.bsimm.com/

- Launched in 2008
- Compares secure development initiatives
- 51 organizations shared data in 2012
- 4 Domains, 12 Practices, 111 activities

BSIMM Structure
4 Domains – 12 Practices

Governance	Intelligence	SSDLC Touchpoints	Deployment
Strategy & Metrics	Attack Models	Architecture & Analysis	Penetration Testing
Compliance & Policy	Security Features & Design	Code Review	Software Environment
Training	Standards & Requirements	Security Testing	Configuration & Vulnerability Management

OpenSAMM

Open Software Assurance Maturity Model
http://www.opensamm.org/

- Created by Fortify Software, now open source
- Managed by **OWASP** http://www.owasp.org
- 4 Business Functions contain 12 Security Practices
- Within each Security Practice, examples are given for increasing levels of program maturity

OpenSAMM Structure

4 Business Functions – 12 Security Practices

Governance	Construction	Verification	Deployment
Strategy & Metrics	Threat Assessment	Design Review	Vulnerability Management
Policy & Compliance	Security Requirements	Code Review	Environment Hardening
Education & Guidance	Secure Architecture	Security Testing	Operational Enablement

BSIMM vs. OpenSAMM

- Both offer resources to build program roadmaps
- 4 "Business Functions" and underlying practices organized more intuitively than 4 "Domains"
- BSIMM's 111 activities more granular regarding controls than OpenSAMM
- OpenSAMM makes it easier to gauge maturity of specific Security Practice areas against the model
- BSIMM provides benchmarks from 12 verticals, may be better for comparison against industry norms

Key Takeaways

- AppSec requires a holistic, programmatic approach
- BSIMM & OpenSAMM: 2 CMMs with similar structures
- Both models contain valuable recommendations
- No need to implement every individual control
- Key is to establish a functioning security ecosystem
- Customize a framework using components from both

BENEFITS OF CAPABILITY MATURITY

Application Security Maturity Value Propositions

- Brand Integrity
- More robust applications
- Reduced risk of breach
- Fewer downstream costs
- Compliance simplified
- Security as a selling point
- Increased business agility

GUIDING CONCEPTS

BSIMM & OpenSAMM are great but…

- Both can be a bit overwhelming
- Which parts will have most impact?
- Start with the 40,000 ft. view

Guiding Concepts for CMM

- Ownership & Accountability
- Application Security Program as Ecosystem
- Socialization & Training
- Risk-oriented Focus
- Open Process Interfaces
- Nurturing Feedback Loops
- Meaningful Metrics
- Automation

Ownership & Accountability

- Executive sponsorship for AppSec program
- Oversight group given real authority
- Program goals made explicit, measurable
- Policies & Standards formalized
- Risk appetite defined & enforced by management
- Individuals know how their roles fit in to big picture
- Processes monitored & refined

AppSec Is a Complex Ecosystem

People

- Customers
- Developers
- Product / Project Managers
- Security Team
- Operations Team
- QA Testers
- Marketing
- Vendors
- Risk/Audit/ Compliance Teams
- Executive Management

Technologies

- SAST
- DAST
- Bug Tracking Systems
- QA Tools
- WAFs
- IPS / IDS / SIEMs
- Application Stack
- Third party code or APIs
- GRC Platform
- SaaS, PaaS, or IaaS

Processes

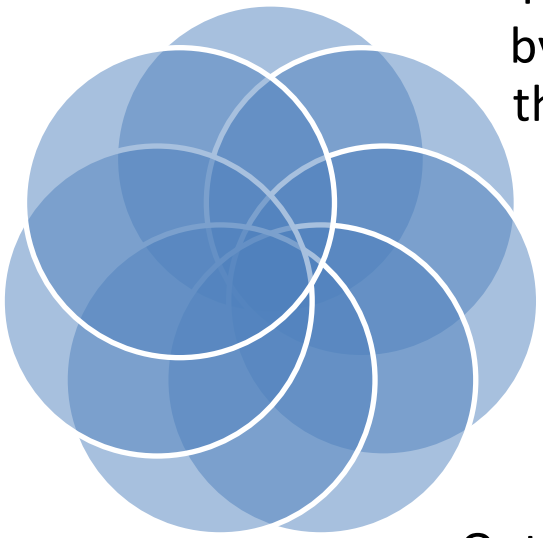
- Governance
- Design & Architecture
- Application Development
- Vulnerability Management
- Threat Modeling
- QA
- Compliance
- Vendor Management
- Operations
- Training
- Project Management

Application Security as Ecosystem

Multiple internal & external stakeholders, dependencies, interfaces

State is healthy & sustainable or *not*

Process influenced by internal/external threats & pressures



Impacts of (perceived) competing interests

Many opportunities to introduce risk

Changes have cascading effects

Output of each process is input for next one

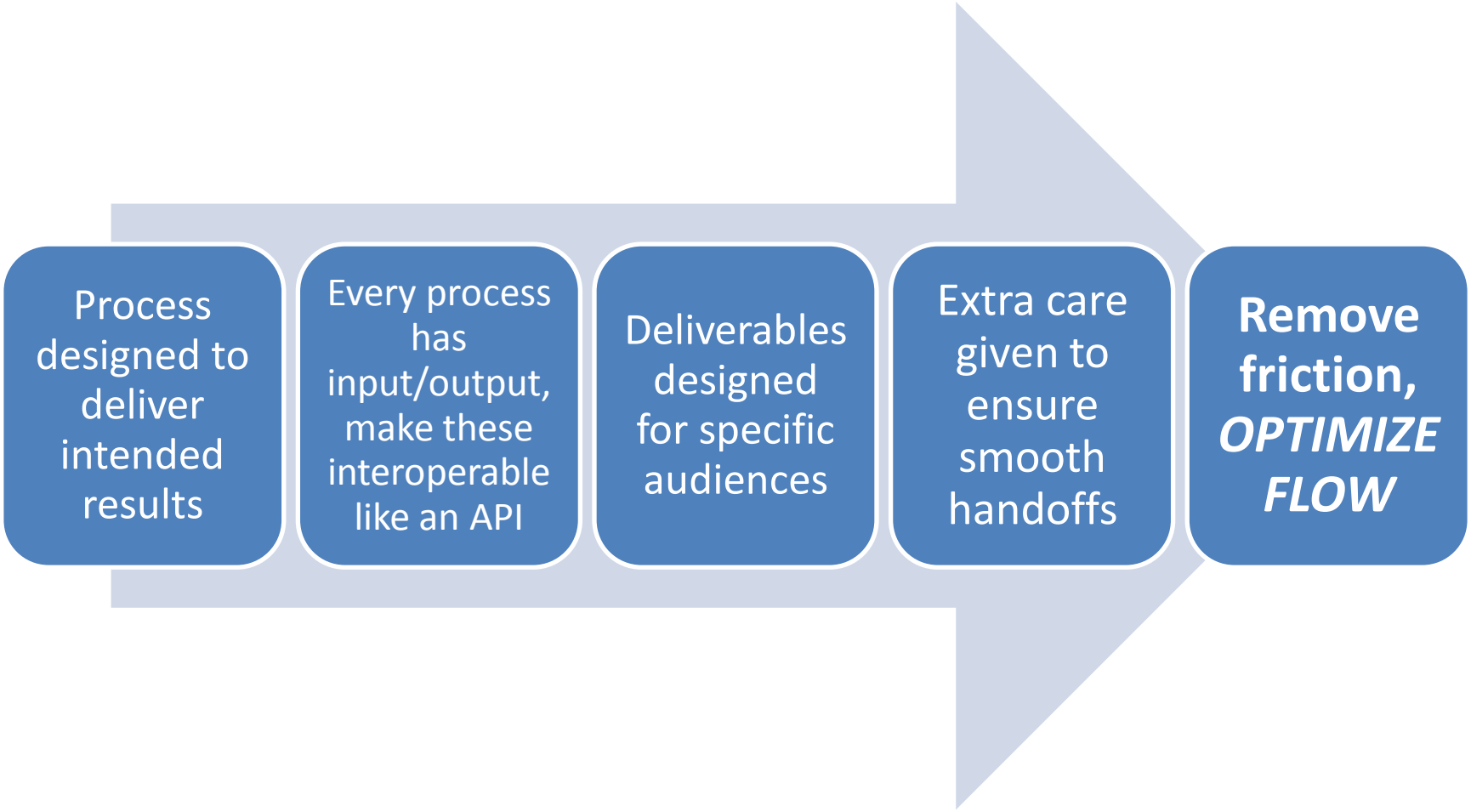
Socialization & Training

- Training provides greatest ROI
- Communication is critical
- Program goals are primary focus
- Risk appetite clearly defined & enforced
- Targeted training for all stakeholders
- Culture of continuous improvement

Risk-Oriented Focus

- Management defines risk appetite
- Understand likely threats to apps
- Know you can't fix everything
- High risk apps require most attention
- Define security/compliance in requirements
- Map vulnerabilities back to actual risk
- Prioritize remediation by risk

Open Process Interfaces



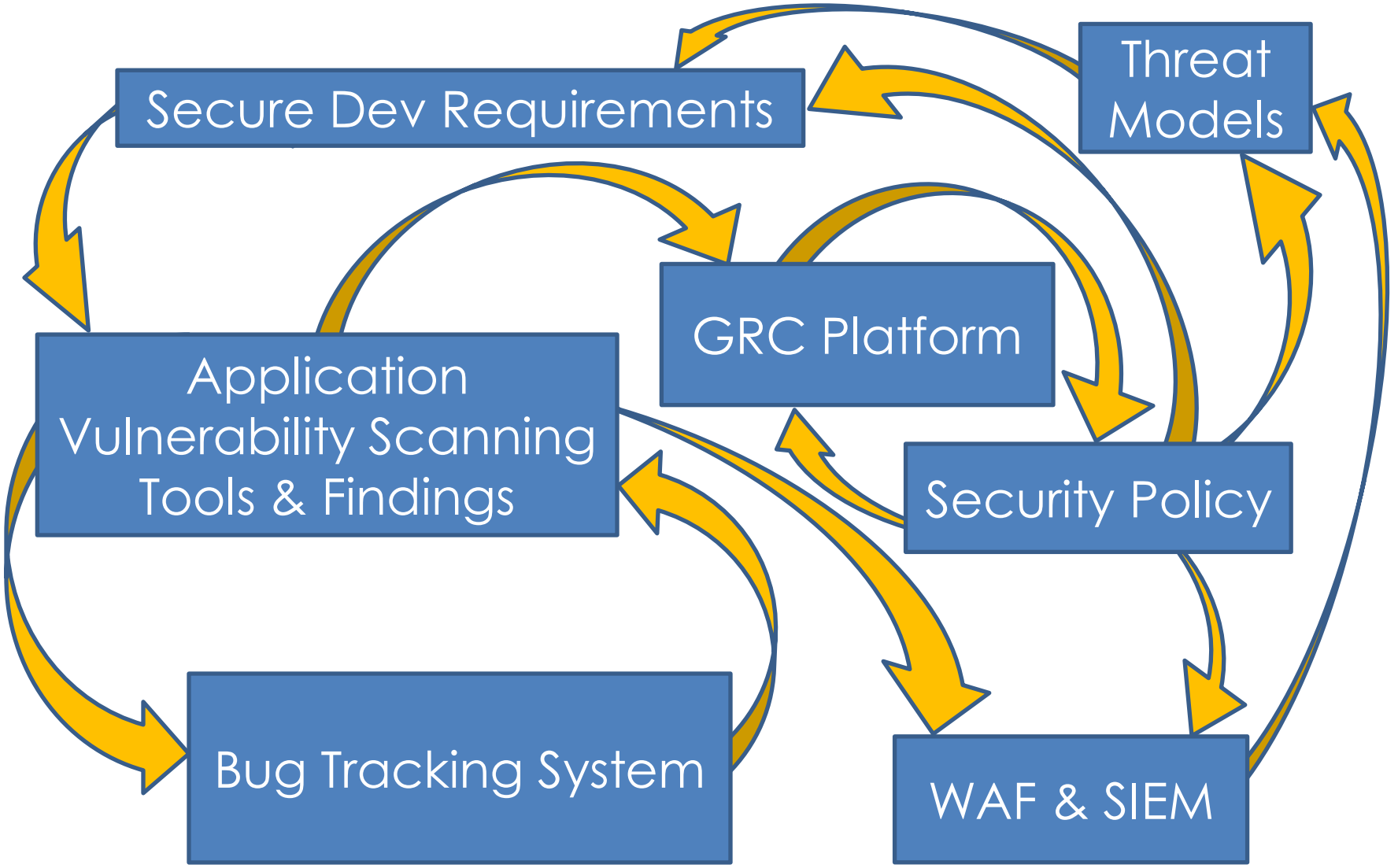
Nurturing Feedback Loops

- Security/Compliance Policies
- Project Management Model
- Design Requirements
- Development Guidelines
- Vulnerability Assessments
- Bug Tracking
- Solution/Tool Configuration
- Continuous monitoring
- Rinse & Repeat

Understand All of These in Terms of:

- Workflows
- Required Inputs
- Downstream Recipients
- Output Usable?
- Lost in transit?
- Obstacles

AppSec Feedback Loop



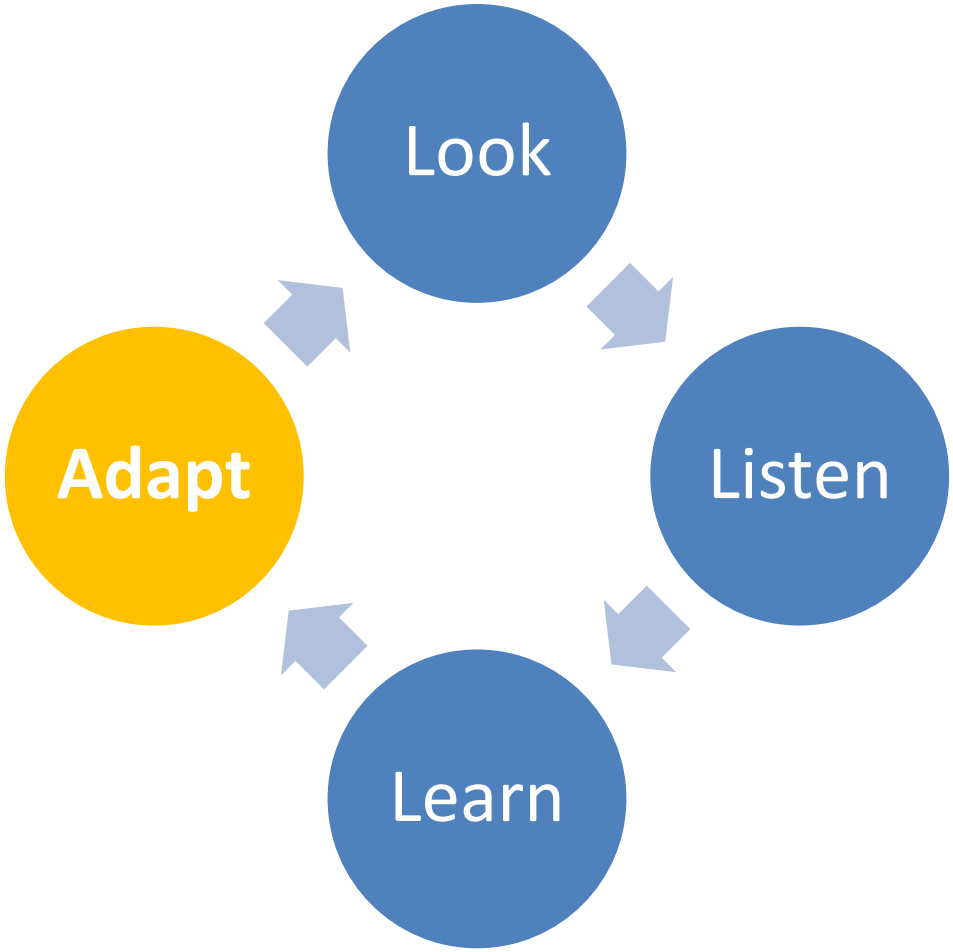
Meaningful Metrics

- If you can't measure it, you can't manage it
- Use metrics to define success
- Are metrics audience appropriate?
- Metrics should pivot up & down the org-chart
- Analyze trends
- Start small, then build

Automation

- Increased volume, accuracy, & consistency
- Frees up bandwidth to focus on strategy
- 1st Step is understanding processes well
- Garbage in, garbage out
- Start small, then build

Build a Program to Evolve



KEY AREAS OF FOCUS

This All Sounds Complicated…

Where do I begin?

Take Inventory

- What do you have?
- How does your program compare to CMM?
- Focus on Risk: all apps are NOT created equal
- Identify High Risk Applications
  (PII, PAN, PHI, $ transactions, high visibility)
- Which compliance standards must be met?
  (GLBA, SOX, FFIEC, FINRA, PCI, HIPAA, ISO, etc.)
- Categorize applications into risk buckets

Key Areas of Focus

- Metrics
- Training
- Controls
- Review
- Testing
- Environment
- Defect Handling

Metrics

- Collected Details
- Trends
- Audiences
  - Internal Team
  - Executives
  - Customers

Training

- Targeted Training
  - Developers
  - Analysts
  - Security Team
  - Executives
- Relevant Content
- Dated

Controls

- Corporate Policies
  - Data Classification
- Compliance
  - PCI, HIPAA, SOX, GLBA, HITECH, ISO, NIST
- Guidance
  - Requirements
  - Standards

Review

- Requirements Review
- Design Review
- Architecture Review
- Peer Review
- Code Review

Testing

- When does testing occur in the SDL?
  - Developer Tools
- Various Tools
  - SAST
  - DAST
  - Pen-testing

Environment

- Staged Deployment
- Deploy to a Secure Environment
- Repetitive Hardening
- Stay Current

Defect Handling

- Security Defect Tracking
- Prioritization
- Verification
- Remediation
- Risk Assumption

5 Key Takeaways

- Test Early and Test Often
- Use Familiar Channels of Communication
- Establish Metrics Early
- Govern the SDL Process
- Train Train Train

Q &A

- Application Security Consulting
- Application Vulnerability Discovery
- Secure Development Training
- Vulnerability Remediation
- Security Architecture
- Secure Development Lifecycle Consulting

www.astechconsulting.com

Thank You

AsTech Consulting
info@astechconsulting.com
http://www.astechconsulting.com

Jonathan Davis
jonathan.davis@astechconsulting.com

Steve Wolf
steve.wolf@astechconsulting.com