LAB3R12 A Successful Application Security Program - Envision Build and Scale

Post Conference Summary

Jyothi Charyulu
Jaya Chilakamarri
# Table of Contents

- **INTRODUCTION** ......................................................................................................................... 2
- **KEY TAKEAWAYS** ..................................................................................................................... 4
- **SUMMARY OF ACTIVITIES** ......................................................................................................... 5
Introduction

Thank you for attending our Learning Lab!

We are very grateful and honored that you chose to spend some of your valuable time attending our Learning Lab, *A successful application security program: Envision, build and scale*, at RSA Conference 2017. There were so many great presentations and tracks, and your participation in our Lab means a lot to us. We hope you found it as engaging and educational as we did, and that it gave you everything you hoped it would.

It is very important to clearly know ‘WHY’ you do something. Why helps you identify your purpose behind the task. When WHY is strong, HOW and WHAT get easy. The most interesting discovery in the breach patterns to industry matrix was the rise of Web App attacks across the board, but especially for financial services organizations (up from 31% in the 2015 DBIR). It is very important that organizations build a robust enterprise application security program that is tightly integrated with the SDLC.

This summary is designed to recap the discussions that made up our time together in the Learning Lab. The top five maxims of application security program are complex and varied, and two hours was barely enough time to scratch the surface. That being said, we felt that all participants both contributed and took away key insights that will help them improve and mature their application security programs and their overall enterprise security posture. That includes both of us, and we couldn’t have done any of it without the enthusiastic and insightful participation of each and every Lab participant.

Everyone should be proud of the work we did at the conference, and we hope that the rest of your RSA Conference experience over the week was as rewarding for you as collaborating on this Lab was for us. Until next year! Thank you again.

Jyothi Charyulu and Jaya Chilakamarri
**Lab Summary**

**Key Takeaways**

**HERE ARE THE TOP 3 CHALLENGES WE HAVE COME ACROSS**

1. Alignment of organization measures and incentives – how do lessen the gap of grief between the business and IT Security risk?
2. Typically, within organizations, we have a Build Culture vs Measure Culture. Software development teams have a build culture. They focus on features, functionality and timelines. Security has a measure culture to ensure secure services are being deployed. How do you bridge this gap?
3. Integration of security across SDLC – think of security up front and make it inherent and not an afterthought.

**THE SOLUTION IS TO INTEGRATE SECURITY WITHIN SDLC:**

Top five maxims for a successful application security program:

1. Program roadmap
2. Leadership mandate
3. Team empowerment
4. Secure SDLC
5. Governance

For a detailed explanation, please review the [presentation](#) at the Learning Lab.
Activities:

- Draw a large rectangle on your whiteboard
- Each member can individually brainstorm on sticky notes – one idea per note that you consider ‘in’ scope for the application security program.

Roles:
- Facilitator:
  - Ensures all participate
  - Promotes discussion
  - Provides feedback at the end of the exercise
- Timekeeper:
  - Monitors time

Benefits and Costs:

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low/High</td>
<td>High/High</td>
</tr>
<tr>
<td>Low/Low</td>
<td>High/Low</td>
</tr>
</tbody>
</table>

Roles and responsibilities:

Upon successful implementation:
Behaviors: More of/Less of
- More of:
  - What will we hear and see more?
- Less of:
  - What will we hear and see less?
A successful application security program: Envision, build and scale
A successful application security program: Envision, build and scale
A successful application security program: Envision, build and scale
Thank you again!

Best regards,

Jyothi Charyulu and Jaya Chilakamarri
A Successful Application Security Program: Envision, Build and Scale

Jyothi Charyulu
Sr. Principal application security architect
Sabre
@sg02411

Jaya Chilakamarri
Director, Enterprise governance
Sabre
@JayChil4
Agenda

- Leadership
- Secure SDLC
- Governance
- Teamwork
- Security Roadmap
Why application security?

40% of attacks via public facing web apps

Targeting credential theft
Point-of-sale attacks continue
89% of breaches had a financial or espionage motive
Hacking, malware distribution, phishing, with social engineering attacks

Ransomware on the rise
Insider threat remains strong
Malware automation

Source: 2016 Verizon data breach report summary
Why application security?

1. Web applications – front door of corporations
2. Apps are top attack vector
3. Breaches are VERY expensive
Application security program – top 3 questions to ask

- What is your current status?
- What is your application security strategy?
- What is your proactive security program?
## Application security – current state

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>38%</td>
<td>• Have ‘maturing’ app sec programs</td>
</tr>
<tr>
<td>67%</td>
<td>• Have partially integrated app sec into overall security</td>
</tr>
<tr>
<td>40%</td>
<td>• Have documented practices for 3\textsuperscript{rd} party vendors</td>
</tr>
<tr>
<td>23%</td>
<td>• Report that applications are the source of breaches, attacks and sensitive data leak</td>
</tr>
<tr>
<td>41%</td>
<td>• Name public facing web applications as leading cause of breaches</td>
</tr>
</tbody>
</table>

Source: Veracode.com
How do you solve the top 3 challenges in security?

- Alignment of organizational measures and incentives
- Prioritization of competitive landscape versus secure services
- Integration of security across SDLC/PLC
By integrating security across organization...

Security checkpoints in product lifecycle from ideation to retirement

Enterprise portfolio management and governance
- Integration training
- Security awareness
- Capture and scope security requirements
- Risk assessment

IT service management
- Ongoing security monitoring and support
- Manual penetration checks
- Continuous training

Security checkpoints across development, quality assurance, metrics, reporting, training
Maxim 1:
Application security program roadmap
Maxim 2: Leadership mandate
Maxim 3: Team empowerment
Maxim 4: Secure SDLC
Maxim 5: Governance
Application security program – top five maxims

1. Program roadmap
2. Leadership mandate
3. Team empowerment
4. Secure SDLC
5. Governance

Envision . . . Build . . . Scale
Application security program roadmap
Envision your P E R F E C T application security program

Things to consider when creating your vision

- What problem are you trying to solve?
- What applications?
- What components – third party/open source/vendors?
- Product lines?
- Extent of impact?
- Which people?
- Other constraints?
Application security program - activity

- Draw a large rectangle on your whiteboard
- Each member can individually brainstorm on sticky notes – one idea per note that you consider ‘in’ scope for the application security program.

- **Roles:**
  - Facilitator:
    - Ensures all participate
    - Promotes discussion
    - Provides feedback at the end of the exercise
  - Timekeeper:
    - Monitors time

- **In & Out of the Frame Exercise**
- **Define YOUR security program definition in 140 characters (Tweet it!)**
Help teams create secure code and reduce the number of vulnerabilities by building sustainable proactive security practices embedded within our PLC and SDLC

134 characters! 😊
application security program vision

- SDLC integration
- Security integrated in design/architecture phase
- Threat modeled and risk ranked applications

Application security inventory:

- Applications
  - Web applications (tier 1) / Client Server (tier 2, tier 3, tierN)
  - Mobile applications / Cloud deployment

- Components
  - 3rd party
  - Vendor
  - Open source
Application security program vision

- Multiple testing techniques:
  - Static Testing (SAST)
  - Dynamic Testing (DAST)
  - Manual Penetration Testing (MPT)
- Risk-based policies
- Training and security awareness
- Remediation
Application security program roadmap

**Envision**

- Know what you need to protect – define scope
- Strategic roadmap – aligned with organizational goals
- Application risk ranking

**Execution** – people, process and technology

- SAST, DAST, WAF, OSS and penetration testing
- Governance – policies, guidelines, standards

*SAST: Static Application Security Testing  
DAST: Dynamic Application Security Testing  
WAF: Web Application Firewalls  
OSS: Open Source Software*
Application security program – leadership mandate

Leadership mandate
Leadership mandate

- Find innovators and early adopters
- Start small – pilot groups
- Establish success in pilots
Leadership mandate

- Top down + bottom up collaboration
- Technical + cultural strategy + leadership
- 15-word program definition
Application security program - activity

Why app sec?

Payoff matrix

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low/High</td>
<td>High/High</td>
</tr>
<tr>
<td>Low/Low</td>
<td>High/Low</td>
</tr>
</tbody>
</table>
Leadership mandate

Adapted from The adoption life cycle by Geoffrey A. Moore in ‘Crossing the Chasm’
Leadership mandate

Crossing the Chasm

- Between early adopters and early majority – there is a large chasm
- Most innovation fails to cross the chasm
- So how do you cross the chasm?
How do you cross the chasm?

Team empowerment  Secure SDLC  Governance
Team empowerment
Team empowerment

Build Value Proposition

- Build matrix of champions
- Change culture not just technology
- Empower development teams
- Define E2E process pipeline
- Drive ownership, accountability
- Continuous training tied with performance measures

‘Little fish learn to be big fish in little ponds’ – Peter Drucker
Empowerment Culture

Security is everyone’s job

- Collaboration – generalists, no specialists
- Autonomous teams and leader boards
- Build Security in day to day job – not just for the app sec team
Key players

Technology Executive

Development

Product Owner

Risk and Security

QA

Release Manager
Application security program - activity

Roles and responsibilities

Role play

Upon successful implementation:
Behaviors: More of/Less of
• More of:
  • What will we hear and see more?
• Less of:
  • What will we hear and see less?
SS DLC – Secure Software Development Lifecycle

Secure SDLC
Secure SDLC

Security that is INVISIBLE

- Security checklists at SDLC gates
- Architecture and code reviews
- Automate security + build

- Integrate with release management
- Automate security + bug tracking
- Continuous communication with metrics
Secure SDLC Key Takeaways

- Security, operations and development teams share accountability
- Create a comprehensive set of governance rules and uniform policies
- Conduct assessments throughout the SDLC
- Automate testing
- Integrate remediation of vulnerabilities into the development process
Secure SDLC – recommendations

Security that is INVISIBLE

- Download and read OWASP ASVS Standard
- Utilize security checks to test applications against
  - OWASP top 10 flaws
  - PCI DSS vulnerabilities
- Complete a OpenSAMM maturity assessment to identify program improvement areas
- Look up games:
  - OWASP cornucopia
  - Elevation of privilege (Microsoft)
Governance
Governance

You are what you measure

- Implement Governance Framework
  - Policy definition
  - Education
  - Metrics
  - Feedback

- Set clear expectations and ground rules to cross-functional teams
Governance

- Create standards and policies based on
  - Cross-functional requirements
  - Industry Standards: OWASP Top 10/PCI/SANS 25/HIPAA
- Weigh remediation versus mitigation
- Measure results
  - Dashboards, leaderboards, reports and audits/assessments
- Update policies as needed
- Continuous training and awareness
Grow to new heights of success

- Leadership
- Roadmap
- Teamwork
- Secure SDLC
- Governance
- Secure Applications
Socialize YOUR app sec program:
What is your program about?
Why is it important?
What does success look like?
What do we need from you?
Apply the maxims

SECURITY that is INVISIBLE

- **Short term goals:**
  - 15-word project definition
  - Collaborate with leadership and build enablement strategy

- **Mid term goals:**
  - Risk-rank applications
  - Build a roadmap for enterprise application security program

- **Long term goals:**
  - Pilot a small subset of applications
  - Integrate with SDLC – toll gates, metrics, portal and communication
Parting wisdom

- “Change is hardest at the beginning, messiest in the middle and best at the end” - Robin Sharma

- No idea works until YOU do the work

- See problems as opportunities for growth

- Caliber of your practice = quality of your performance

- Now – go and conquer your app sec world!
Thank you!

Questions?

Jyothi Charyulu: Jyothi.Charyulu@sabre.com
Jaya Chilakamarri: Jaya.Chilakamarri@sabre.com
Secure design principles

Plan the application architecture, technology stack, and investigate requirements including:

- **External dependencies**: Interfaces, libraries, frameworks, hardware, and software that the application is dependent upon
- **Entry Points**: The services through which an attacker can interact with the application or supply it with data
- **Assets**: Physical and abstract data and information
- **Trust levels**: The level of access/permissions granted by the application to external entities
Secure design principles

Plan security requirements based on:

- Authentication and session management
- Access control
- Input validation and output encoding
- Cryptography and data protection
- Error handling and logging
- Communication and HTTP Security
- Files and resources
# NIST 500-269

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Related Terms</th>
<th>OWASP Top Ten 2007</th>
<th>CWE ID</th>
<th>PCI DSS 1.1</th>
<th>Insecure Direct Object Reference</th>
<th>Unvalidated input is used as a reference to an internal implementation object, such as a file, directory, or database key.</th>
<th>Parameter tampering, Cookie poisoning, Path manipulation</th>
<th>A4</th>
<th>A1</th>
<th>79</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross Site Scripting (XSS)</td>
<td>A web application accepts user input (such as client-side scripts and hyperlinks to an attacker's site) and displays it within its generated web pages without proper validation.</td>
<td>Reflected XSS, persistent (stored) XSS, DOM-based XSS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Cross Site Request Forgery (CSRF)</td>
<td>An application authorizes requests based only on credentials that are automatically submitted by the browser. A CSRF attack forces a logged-in victim's browser to send a request to a vulnerable application, which then performs the chosen action on behalf of the victim, to the benefit of the attacker.</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SQL Injection</td>
<td>Unvalidated input is used in construction of an SQL statement.</td>
<td>Blind SQL injection</td>
<td>A2</td>
<td>89</td>
<td>X</td>
<td></td>
<td>Information Leakage</td>
<td>Disclosure of sensitive information or the internal details of the application.</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>OS Command Injection</td>
<td>Unvalidated input is used in an argument to a system operation execution function.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A5</td>
<td></td>
<td>352</td>
<td></td>
</tr>
<tr>
<td>XML Injection</td>
<td>Unvalidated input is inserted into an XML document.</td>
<td>XPath injection, XQuery injection</td>
<td>A2</td>
<td>91</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>HTTP Response Splitting</td>
<td>Unvalidated input is used in construction of HTTP response headers.</td>
<td>CRLE injection</td>
<td>A2</td>
<td>113, 93</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Malicious File Inclusion</td>
<td>Unvalidated input is used in an argument to file or stream functions.</td>
<td>File inclusion, Remote code execution, Directory traversal</td>
<td>A3</td>
<td>98</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

For more information, visit [https://samate.nist.gov/docs/webapp_scanner_spec_sp500-269.pdf](https://samate.nist.gov/docs/webapp_scanner_spec_sp500-269.pdf)
Additional resources for secure code

- PCI SSC (Payment Card Industry Security Standards Council)
- OWASP (Open Web Application Security Project)
- OWASP ASVS: Application Security Verification Standard 3.0.1
- OWASP Guidelines – cheat sheet on SQL Injection Prevention
- OWASP Code Review Guide: Reviewing code for OS Injection
- OWASP Development Guide: Buffer Overflows
- OWASP Cryptographic Storage Cheat Sheet
- OWASP Guide to Cryptography
- OWASP Transport Layer Protection Cheat Sheet
Additional resources for secure code

- OpenSAMM
- WASC (Web Application Security Consortium)
- SANS institute
- CWE/SANS top 25 most dangerous software errors
- The Security Content Automation Protocol (SCAP)
- iOS mobile development guidelines
- Android mobile development guidelines
- Heartbleed bug
- ShellShock bug