Mean Time to Fix (MTTF)
IT Risk’s Dirty Little Secret

Joe Krull, CPP, CISSP, IAM, CISA, A.Inst.ISP, CRISC, CIPP
Presentation Overview

- Basic Application Security (AppSec) Fundamentals
- Risks Associated With Vulnerable Applications
- Understanding the Software Attack Surface
- “Mean Time to Fix” (MTTF) Explained
- Application Testing – More Than Simple Scans
- How to Include AppSec in Annual Risk Assessments
- Tools and Resources to Assess and Audit AppSec Maturity
Application Security Fundamentals

- Application security includes measures taken throughout an application's life-cycle to prevent exceptions in the security policy of an application or the underlying system (vulnerabilities) through flaws in the design, development, deployment, upgrade, or maintenance of the application.*

- The primary focus is on Layer 7 of the OSI Model

- AppSec should be part of an organization’s or vendor’s Software (or System) Development Life-Cycle (SDLC)

- A key component of application security should be for developers and their managers to be aware of basic AppSec requirements, common threats and effective countermeasures

- AppSec knowledge and maturity is significantly lower today than traditional network security

* Wikipedia
Risks Associated With Vulnerable Applications

- Unauthorized access to sensitive customer or company data
- Theft of sensitive data to conduct identity theft, credit card fraud or other crimes
- Defacement of websites; strong potential for brand damage
- Manipulation of data impacting data integrity, quality and organization’s reputation
- Redirection of users to malicious web sites; phishing and malware distribution
- Denial of service; availability of data
- Attackers can assume valid user identities
- Access to hidden web pages using forged URLs
- Attacker’s hostile data can trick the interpreter to execute unintended commands
What Is Your Software Attack Surface?

To assess application security, many organizations focus on obvious software resources, but overlook their overall inventory of applications and code from less obvious sources when they analyze their assets.

Software You Currently Know About

What’s Normally In This Category?
• Critical legacy systems
• Notable web applications

Why Do These Usually Merit Consideration?
• Lots of monetary or brand value flows through them
• Compliance requirements (e.g. PCI, HIPAA, FFIEC, etc.)
• Formal SLAs with customers
• You’ve had one or more previous security incidents (or near misses)
What Is Your Software Attack Surface – Part 2?

Add In the Rest of the Web Applications Your Organization Actually Develops and Maintains

What’s Normally in This Category?
• Line of business applications
• Event-specific applications, e.g. holiday apps, sales support, open enrollments

Why Could You Miss Them in Your Analysis?
• Lack of knowledge, overlooked or forgot they were there
• Line of business procured through non-standard channels
• Added through a merger or acquisition
• Believed to be retired but still active
What Is Your Software Attack Surface – Part 3?

Add In the Software You Bought from Somewhere

What’s Normally in This Category?
• Less known or utilized line of business applications
• Support applications
• Infrastructure applications

Why Could You Miss Them in Your Analysis?
• Automated scanners are good at finding web applications. Non-web, not so much.
• Contract language or un-validated assumptions that the application vendor has security “covered”
What Is Your Software Attack Surface – Part 4?

What’s Normally in This Category?
- Support for line of business functions
- General marketing and promotion
- Financial analysis applications
- Software as a Service (SaaS)
- Mobile applications
- User procured software

Why Could You Miss Them in Your Analysis?
- Decentralized procurement
- Ineffective security policies
- Use of prohibited software
- Lack of awareness
As perception of the problem of attack surface grows, the scope of the problem increases – or, the more you know, the more you need to assess.
Value and Risk Are Not Equally Distributed

• Some Applications Matter More Than Others
  – Value and character of data being managed
  – Value of the transactions being processed
  – Cost of downtime and breaches

• Therefore All Applications Should Not Be Treated the Same
  – Allocate different levels of resources to assurance
  – Select different assurance activities
  – Also must often address compliance and regulatory requirements
Myth #1 – I Don’t Need AppSec Because My Network is Secure

Technical Rationale

Non-Technical Rationale
Mean Time to Fix (MTTF)

- A 2013 industry study from White Hat Security revealed that the “Mean Time to Fix” for web application flaws categorized as “serious” averaged 193 days across all industries.
- In the same study, for one industry (Education) the figure jumped to 342 days of exposure.
- In a similar study from Veracode, 70% of 22,430 applications submitted to their testing platform in 2012 contained exploitable security vulnerabilities.
- How would you report to your management that a “serious” and likely exploitable vulnerability was present on your primary public facing web site or a 3rd party hosted portal for more than six months?
- What compensating control or controls do you think you could explain to placate management that a serious vulnerability could not be exploited?
- Verizon’s 2013 Breach Report says 90% of attacks last year were perpetrated by outsiders and 52% used some form of hacking. How does this help you explain application risk?
Myth #2 – An Automated Scanner Can Find All The Application Vulnerabilities That Exist

- There is no “silver bullet” for identifying application security vulnerabilities. There are different classes of tools ranging from static code scanners that assess the code to dynamic scanners that analyze logic and data flow. Generally, 30% to 40% of vulnerabilities can be identified by scanners; the remainder are uncovered by other means.
- Manual testing allows an informed and experienced tester to attempt to manipulate the application, escalate privileges or get the application to operate in a way it was not designed to do.
- But wait, there’s more…………
What Goes Into An Application Test?

Application security goes well beyond simply running a scanning tool. For critical or high value applications, or those that process sensitive data, thorough testing may actually include a combination of several methods.
AppSec – What Can You Do and Why?

Information Security Professionals

- Promote AppSec awareness in your organization
- Confirm that application security testing is part of your overall security program
- Demand that all applications developed by 3rd parties be tested and remediated prior to being placed in production
- Get all developers and their managers trained on AppSec
- Obtain and review the SDLC from a security perspective

IT Auditors

- Influence your Chief Audit Executive to include AppSec in the organization’s annual risk assessment
- Increase your relevance and value to your organization by identifying risks associated with poorly coded applications
- Conduct a simple initial audit to assess what controls are in place
- Conduct a subsequent audit to determine the effectiveness of those controls; measure MTTF
Tools and Resources

- Open Software Assurance Maturity Model (OpenSAMM) – A freely available open source framework that organizations can use to build and assess their software security programs
  www.opensamm.org

- The Open Web Application Security Project (OWASP) – Worldwide not-for-profit organization focused on improving the security of software. Source of valuable free resources
  www.owasp.org

- Open Source or Low Cost Application Security Scanners – OWASP Zed Attack Proxy (ZAP), w3af, Mavituna Netsparker, Websecurify, Wapiti, N-Stalker, SkipFish, Scrawlr, Acunetix, and many more to do basic discovery work
The OWASP Top 10 For 2013

• A1 Injection
• A2 Broken Authentication and Session Management
• A3 Cross-Site Scripting (XSS)
• A4 Insecure Direct Object References
• A5 Security Misconfiguration
• A6 Sensitive Data Exposure
• A7 Missing Function Level Access Control
• A8 Cross-Site Request Forgery (CSRF)
• A9 Using Components with Known Vulnerabilities
• A10 Unvalidated Redirects and Forwards

## Example AppSec Audit Work Program

### Software Assurance Maturity Model (SAMM) Scorecard

<table>
<thead>
<tr>
<th>Business Functions</th>
<th>#</th>
<th>Security Practices/Phase</th>
<th>Maturity Level</th>
<th>Level 1 Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>Governance</td>
<td>1</td>
<td>Strategy &amp; Metrics</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Policy &amp; Compliance</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Education &amp; Guidance</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>4</td>
<td>Threat Assessment</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Security Requirements</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Secure Architecture</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>7</td>
<td>Design Review</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>Code Review</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>Security Testing</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Deployment</td>
<td>10</td>
<td>Vulnerability Management</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Environment Hardening</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Operational Enablement</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### SAMM Valid Maturity Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Implicit starting point representing the activities in the Practice being unfulfilled</td>
</tr>
<tr>
<td>1</td>
<td>Initial understanding and ad hoc provision of Security Practice</td>
</tr>
<tr>
<td>2</td>
<td>Increase efficiency and/or effectiveness of the Security Practice</td>
</tr>
<tr>
<td>3</td>
<td>Comprehensive mastery of the Security Practice at scale</td>
</tr>
</tbody>
</table>

**Legend**

- Green: Objective Activity was met.
- Red: Objective Activity was not met.
Open Source Software Vulnerability Management Tool

– Supports consolidation and de-duplication of imported results from scanner tools, manual testing and threat modeling
– Provides reports on application security status and trending over time
– Translates application vulnerabilities into software defects and pushes tasks to developers in the tools and systems they are already using
– Creates virtual Web Application Firewall (WAF) rules to help block malicious traffic while vulnerabilities are being resolved. While your organization takes on remediation of your applications, virtual patching helps guard against common vulnerabilities such as Cross-Site Scripting (XSS) and SQL Injections.
– Compatible with a number of commercial and freely available dynamic and static scanning technologies, SaaS testing platforms, IDS/IPS and WAFs and defect trackers

ThreadFix Auditor

– Currently in Development – Virtual Application Scanner – Will allow audit and security professionals to identify, track and report on application security vulnerabilities and remediation activities/effectiveness
Questions / Contact Information

Joe Krull
Director
jkrull@denimgroup.com
(210) 572-4400

www.denimgroup.com
blog.denimgroup.com