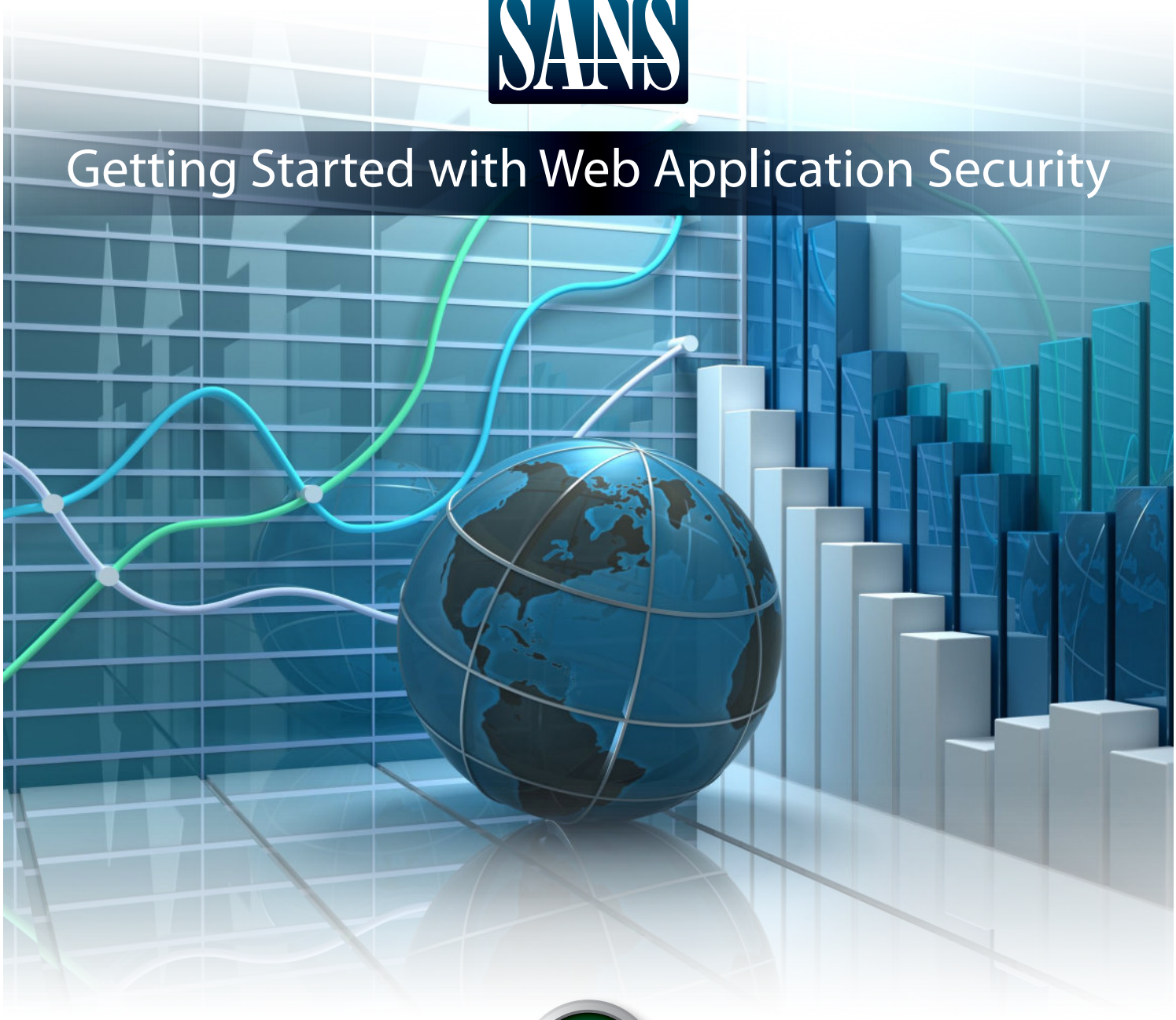

Getting Started with Web Application Security

Written by Gregory Leonard

February 2016

Sponsored by
Veracode

Since as far back as 2005,[1] web applications have been attackers' predominant target for the rich data that can be pulled from them. Attackers also use web applications against customers and to probe deeper into other connected enterprise systems.

Know the Risks

Today, as Figure 1 shows, public-facing web applications are still by far the top concern for developers and managers of applications, according to the 2015 SANS State of Application Security Survey.[2]

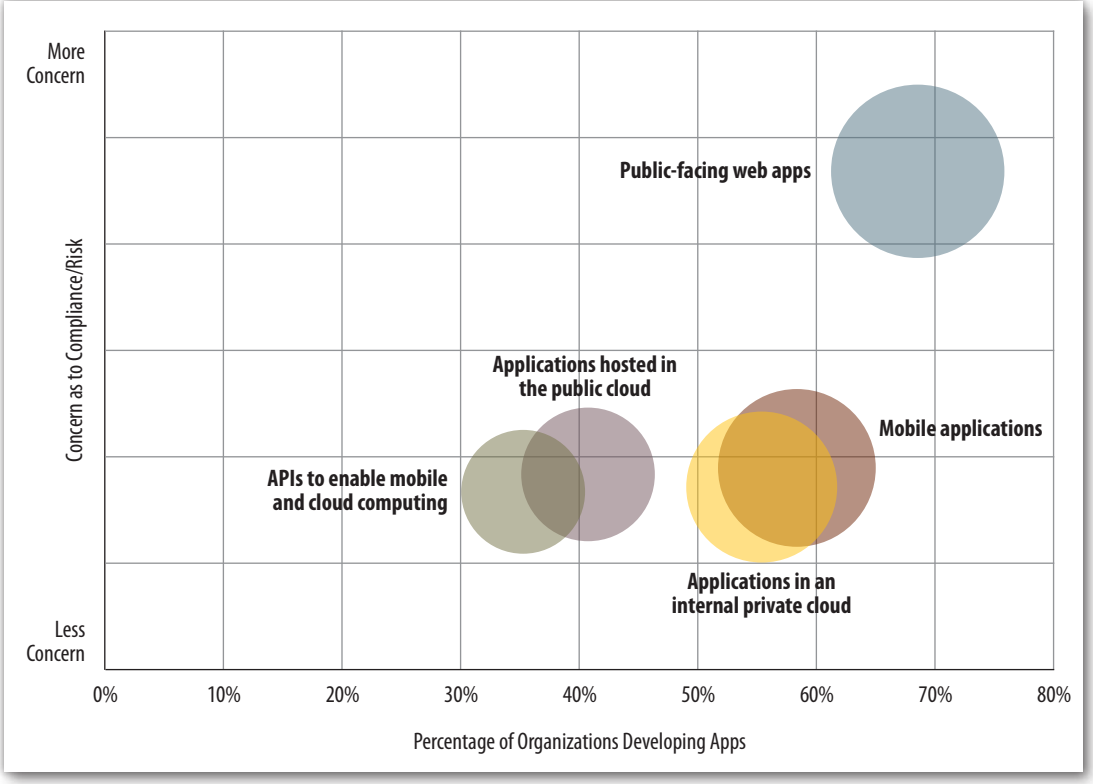

Figure 1. Web Applications Deemed Most Risky

Applications developed from a variety of reusable components and popular frameworks, such as Java and .NET, are rife with vulnerabilities. Vulnerable components, along with insecure development practices, are setting the stage for attackers to easily manipulate vulnerabilities found in code, as well as functional vulnerabilities, such as confusing forms fields on trusted websites. As such, attackers are turning trusted websites into "drive-by" traps to capture access information and transmit malware to unsuspecting users.

Attacks against web applications have become more organized, prolific and polymorphic. For example, they are also using advanced, hard-to-detect distributed denial of service (DDoS) techniques to halt business or confuse responders while they open back doors.

[1] www-07.ibm.com/sg/smarterbusiness/meettheexperts/includes/downloads/Securing_Your_Web_0910_eve.pdf

[2] www.sans.org/reading-room/whitepapers/analyst/2015-state-application-security-closing-gap-35942, Figures 3, 4 and 5

Most organizations today operate on the web, and need to protect their applications, business and users from these and other risks. If you don't already have a program, it's critical to get one started. The key is knowing which applications need protecting, what those applications connect to, and what type of data and access is processed through the applications. Then estimate the value of the data and credentials against the cost of losing control to attackers.

There is no formula to calculate the return on an application security program investment, but the relatively low cost of implementation pales against the potentially high costs of a breach. You must first know where and how to get started with web application security.

Bridge Silos

The first step is to establish a web application security perimeter. This perimeter refers to all Internet-facing applications hosted or managed by your organization. Because there are many people involved in developing and managing these applications, be ready to face cultural and organizational obstacles in trying to define the entire perimeter, including:

- **Inconsistent management of perimeter application knowledge.** Many organizations suffer from the silo mentality,[3] where business departments don't want to share knowledge with one another. These silos result in information gaps, making it more difficult to properly establish the organization's complete perimeter.
- **Organizational gaps between application builders and defenders.**[4] Builders, consisting of developers and their organizations, are primarily focused on the applications they are writing and time to market. They hold essential application security knowledge, such as the sensitive data managed by your applications, that may be targeted by attackers. Defenders, consisting of security and operations teams managing applications in production, are focused on the secure deployment and execution of applications and hold essential knowledge about how your application infrastructure is hardened against attacks. When these two groups don't communicate, defining the web app perimeter becomes more difficult, as both sides may incorrectly assume that security protections are being implemented by the other group.

[3] www.businessdictionary.com/definition/silo-mentality.html

[4] www.sans.org/reading-room/whitepapers/analyst/2015-state-application-security-closing-gap-35942

- **Complexity of the perimeter increased by recent web application development trends.** In the past, web applications were traditionally hosted in private data centers with users connecting via web browsers on desktop computers. Modern web applications, however, have a larger spectrum of hosting choices, including software-as-a-service (SaaS), platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS) options. User connectivity choices have increased as well, with mobile web browsers, native mobile applications and Internet of Things (IoT) devices being added as clients.

If your organization is involved with the acquisition of other companies, consider how the perimeter of acquired companies will be managed. Even if the perimeter management group has been successful in creating a well-defined perimeter for your organization, the following questions need to be addressed to ensure that an acquired company's perimeter is being properly integrated:

- Has the acquired company done its due diligence in creating its own perimeter, or does the perimeter management group need to perform a full analysis?
- Does the acquired company's perimeter match the requirements of your organization, or does it require enhancement?
- How much effort would be necessary to integrate the acquisition's applications into your perimeter?
- How does your perimeter need to be modified to account for the acquisition's applications?

Proper identification of your organization's web applications is complicated by these factors. Additionally, many organizations realize that they don't even know how many applications they have in production. You can use discovery tools to automatically crawl, scan and catalog the IP addresses and domains owned by your organization. These tools not only provide a full mapping of applications that are visible over the Internet but also give your organization confidence that the entire perimeter has been reviewed.

Improve Perimeter Visibility

Before you create your organization's web application security perimeter, you need to address these obstacles to ensure proper security coverage. Break down the silo mentality to allow the sharing of information among teams.[5] Establish a single group that will be responsible for defining and maintaining the web application perimeter for your organization, and make sure that you have the support of management (see sidebar). Without this support, the perimeter management group will have a difficult time overcoming the silo mind-set.

Costly Not to Protect

Although estimating the cost of a data breach based on the number of records stolen is not an ideal metric, it is a good starting point. The 2015 Verizon Data Breach Investigations Report shows that web application breaches exposing millions of records can end up costing an organization hundreds of millions of dollars.[7] In another 2015 report, Ponemon estimated the average cost per year for organizations dealing with attacks on web applications is $3.1 million.[8] Almost 61% of costs were due to a loss of data or disruption of service as a result of the attacks, while remaining costs were associated with preventive measures. And don't discount DDoS attacks. Studies have shown that such attacks can cost organizations $40,000 per hour on average.[9]

This information can help you provide guidance to management when discussing the financial risks of not implementing an application security program. Gaining management support will enable your application security efforts to develop more smoothly. Rather than using vague concerns about what happens if the company experiences a security breach, provide solid cost estimates to empower management to make calculated decisions.

Next, the builders and defenders need to start working more closely together. The IT industry is making significant improvements in this area, as seen in the DevOps movement.[6] As development practices continue moving toward rapid deployment schedules, DevOps becomes a crucial bridge between builders and defenders. If your organization has a DevOps group, it can be used as a resource in supporting your web application security perimeter.

However, if a DevOps group does not exist, your perimeter management group needs to identify an alternative for how this knowledge will be managed. For example, each development team can be given a designated point of contact within the operations group. This point of contact would be responsible for helping all groups understand the development team's application, establishing infrastructure security configuration and ensuring proper perimeter coverage.

[5] www.forbes.com/sites/brentgleeson/2013/10/02/the-silo-mentality-how-to-break-down-the-barriers/

[6] https://en.wikipedia.org/wiki/DevOps

[7] www.verizonenterprise.com/DBIR/2015/

[8] www.stateoftheinternet.com/resources-web-security-white-paper-2015-ponemon-institute-the-cost-of-web-application-attacks.html

[9] www.securityweek.com/ddos-attacks-cost-40000-hour-incapsula

Finally, the perimeter management group needs to consider the type of environment in which your applications are deployed. For cloud-based services (SaaS, PaaS and IaaS), determine who is responsible for specific security controls. Figure 2 provides some perspective on defining a perimeter.

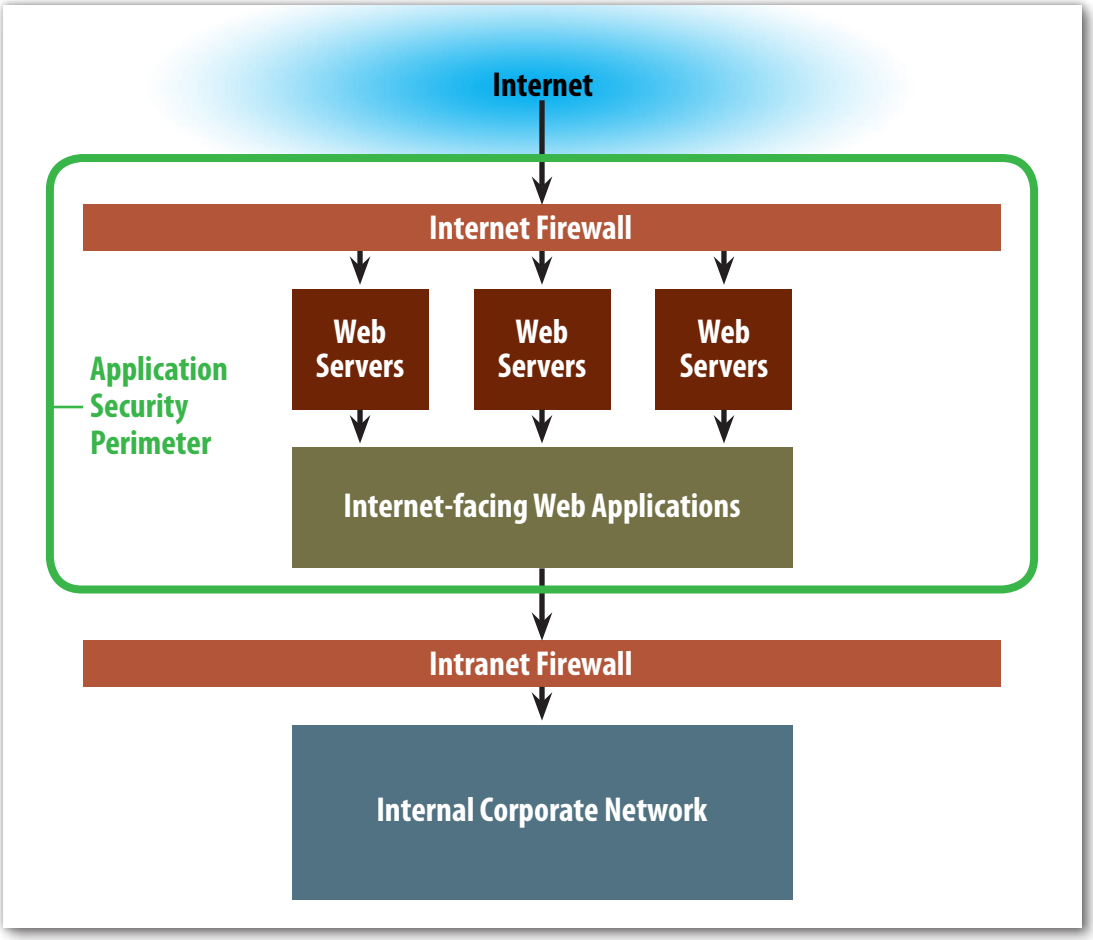

Figure 2. Putting the Perimeter in Perspective

Understand what types of testing are allowed based on the licensing agreements between your organization and your service providers. Additionally, evaluate your application users to determine what types of security testing tools are necessary for sufficient security coverage. For example, is mobile application testing required?

Set Your Road Map

To get control over your web applications, follow these steps:

1. **Start securing your web application security perimeter.** Use an iterative process for securing your perimeter. Your organization will benefit more from securing an incomplete perimeter now than hardening an entire perimeter much later. If your organization is large, or if you have a complex deployment environment, don't wait until 100% of the perimeter data has been collected. Start by defining high-priority objectives first, working with the business units and IT administrators responsible for those applications.

2. **Perform tool selection.** Not all security tools are created equal, nor does any security tool cover all categories of security testing and secure management of operational applications.[10] Evaluate the security risks facing your perimeter and determine which security tool categories fit your organization's needs. Generally, it will be a combination of tools. It is not necessary to select tools from every category, but make sure the tools you choose match your perimeter security requirements.

   For example, traditional security testing tools cover static analysis of source code and dynamic testing of web applications. Newer security testing techniques include blending static and dynamic testing into an interactive form and performing self-protection of applications at runtime. For applications in production, web application vulnerability scanners/vulnerability management systems and firewalls are also useful tools to consider (more detail on all of these tool types is provided in the next section).

3. **Define implementation metrics.** Without measurable statistics, it is difficult to determine when your organization has installed enough of the right security tools where needed, and how effective those tools are. Metrics should include percentage of source code scanned by static analysis tools, coverage of dynamic testing tools over the spectrum of available URLs available for your perimeter applications, and collection of improvements on critical processes for your organization.

Find a balance: Overly aggressive deadlines may result in incomplete or ineffective security tool implementations, while overly lax deadlines will leave your perimeter applications vulnerable for an unacceptable period of time.

[10] http://searchsecurity.techtarget.com/opinion/McGraw-on-why-DAST-and-RASP-arent-enterprise-scale

4. **Set an implementation deadline and checkpoints.** Determine when you want to complete your tools implementation for your perimeter, taking the scale of the implementation into account, and set a deadline. After your deadline has been established, define periodic checkpoints. These checkpoints will be used to determine whether the deadline is in jeopardy of being missed, and will also give your organization the opportunity to incorporate feedback from metrics into the process and update the perimeter based on subsequent analysis.

5. **Evaluate collected metrics and adjust programs based on results.** Defining metrics is only effective if metric data is collected and reviewed. This analysis is the best way to determine whether the security tool implementation your organization has chosen is providing benefit. After all, how else are you going to know that your security perimeter is being protected?

Select Tools

Web application security tools fall under several categories: testing, scanning and protection. Many security tools are designed to scan for well-known vulnerabilities, such as those defined by the Open Web Application Security Project (OWASP) Top 10,[11] the CWE/SANS Top 25,[12] or other requirements related to government or industry security standards. Each category has its own strengths and weaknesses, so evaluate each to see how it fits within your organization. Figure 3 shows which testing tools fit best in the software development lifecycle.

,

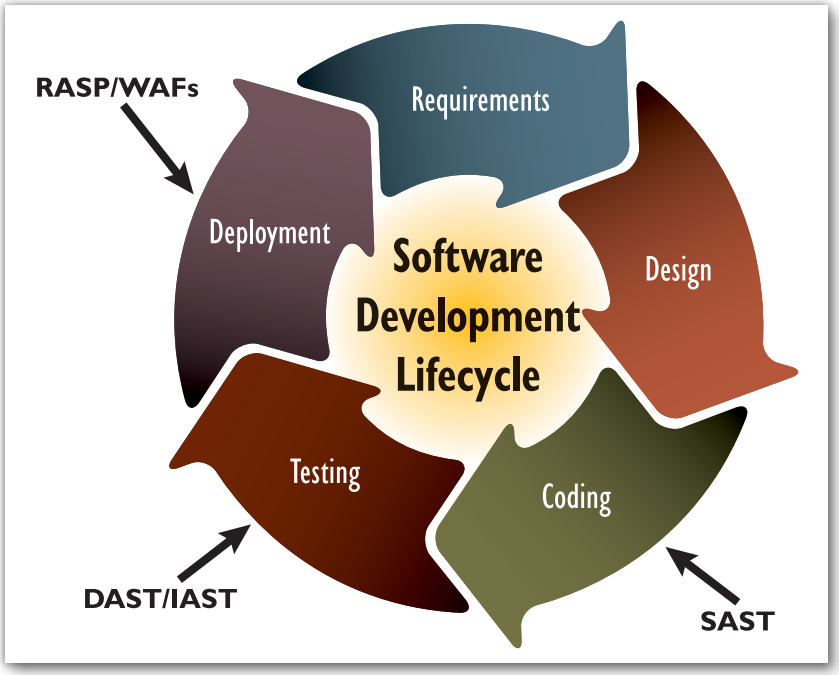

Figure 3. Software Development Lifecycle Aligned with the Most Effective Testing Tools

[11] www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

[12] http://cwe.mitre.org/top25

For Applications in Development

- **Static Application Security Testing (SAST)** – This approach analyzes the source code of an application and compares the scanned code against a database of coding patterns for known security vulnerabilities. When integrated with your organization's automated builds, SAST tools help detect security bugs when they are first delivered to the source code stream, which allows teams to quickly remediate bugs and prevent deployment of builds with vulnerabilities.
- **Dynamic Application Security Testing (DAST)** – This tool uses a black-box testing approach against a running instance of the application being tested. DAST tools use a dictionary of fuzzing techniques and analysis of response data returned by the server to detect vulnerabilities that can be exploited in the running application. Testing with DAST tools provides useful feedback because it approaches the application the way an attacker would.
- **Interactive Application Security Testing (IAST)** – This approach is a combination of SAST and DAST, using instrumentation of the running application's code to scan dynamic test data and get a more accurate determination of the success of the dynamic tests. IAST tools submit fuzzing attacks against your application just like DAST tools, but the instrumentation captures the actual execution flow of the fuzzed data through the application for more accurate results. IAST tools can detect vulnerabilities that do not generate the types of response data that is needed by a DAST tool for detection.

For Applications in Production

- **Runtime Application Self-Protection (RASP)** – This approach involves rewriting application code to manage security tasks within the application instead of delegating events to an external monitoring process. RASP tools provide greater domain knowledge, as the protections are implemented by the application's development teams. This protection could possibly have performance implications if not implemented properly because the application now must process business rules and security functions.
- **Web application firewalls (WAFs)**[13] – These are perimeter devices that scan HTTP requests and response traffic for a web application, and filter out suspected malicious traffic due to some common attacks such as cross-site scripting and SQL injection. They are also designed to scan traffic to "learn" what is acceptable and use this learned information to enhance the whitelist and blacklist validation performed by the WAF.

[13] www.owasp.org/index.php/Web_Application_Firewall

WAFs can be useful when used as part of a defense-in-depth strategy, but it is a mistake to think that all your organization needs is a web application firewall. WAFs catch what they know and have been configured to detect, and are known to miss other indicators of compromise. Configuring WAFs and maintaining them can be problematic and time consuming. For example, new features and application updates will affect application behavior, and the WAF will need to re-learn the application's expected behavior.

- **Vulnerability management** – You want your program to protect the application throughout its life cycle, from inception through retirement. One key tenet of the CIS Critical Security Controls is continuously monitoring for vulnerabilities.[14] As you develop your application security program, plan for the future by combining vulnerability monitoring, firewall and testing data to manage vulnerabilities and respond to events stemming from web applications.

Track Progress

So you have successfully defined your web application security perimeter, you have chosen your security tools, and now you are getting vulnerability reports from your perimeter application scans. Understand that even with all this implemented, you will not have a vulnerability-free perimeter overnight. Your perimeter management group has several tasks it will need to perform on an ongoing basis to ensure that your perimeter protection solution is working effectively. Follow this task list for measuring progress and improving processes as needed:

- **Track milestone and deadline progress.** Tracking milestones will allow your perimeter management group to react quickly to ensure that the deadline does not slip.
- **Aggregate vulnerability data into a single repository.** Most security tools are very good at reporting detected vulnerabilities. Unfortunately, collecting vulnerability data from separate tools is a much more difficult task, as few tools support interfacing with shared repositories. It is, however, important to collect this information to ensure that all vulnerabilities are accounted for. This aggregation also prevents duplicate remediation efforts if more than one tool reports the same vulnerability.
- **Enter vulnerabilities in project tracking tools.** To ensure that security is given the appropriate priority with your development teams, vulnerabilities should be entered as defects in the same project tracking tools as are functional defects and enhancements. The vulnerabilities can then be prioritized alongside the other work items.

[14] www.cisecurity.org/critical-controls.cfm

- **Track metrics over time.** Running security tools is not effective if vulnerabilities are not remediated, so it is important for your perimeter management group to track the number of defects being reported by the security tools. Many security tools can track trends, including whether known vulnerabilities have been remediated and whether previously fixed vulnerabilities have been re-introduced.

If a perimeter application does not show an improving trend for eliminating vulnerabilities, your perimeter management group should collaborate with the responsible development teams to help resolve any issues.

Make the Commitment

Implementing an application security program requires a significant amount of effort over a long period of time, and it may feel like a daunting task to your organization to simply get started. Focusing on the web application security perimeter, a high-risk and high-importance security component, is an excellent starting point. Once the perimeter has been secured, your organization can begin to work on other long-term aspects of a complete application security lifecycle.

About the Author

Gregory Leonard is a co-author and instructor for the SANS DEV541 "Secure Coding in Java/JEE" course and holds the GSSP-Java certification. He has more than 17 years of experience in software development, with an emphasis on writing large-scale enterprise applications. Greg's responsibilities over the course of his career have included ensuring application architecture and security, performing infrastructure design and implementation, providing security analysis, conducting code reviews and evaluating performance diagnostics. He is currently employed as an application security consultant.

Sponsor

SANS would like to thank this paper's sponsor:

VERACODE