SDG
[technology + passion] – risk

- 55 North Water Street
  Norwalk, CT 06854
- 203.866.8886
- sdgc.com

A Strategic Approach to **Web Application Security**

Web application security is a crucial component *of an organization's overall risk management strategy.*

Because of their ubiquity, web applications have emerged as the poster child for cybersecurity threats. Although network and operating system security analysts are constantly on the lookout for new avenues of attack, web applications remain vulnerable. Unfortunately, this susceptibility has given potential hackers an offensive advantage and put web application developers on the defensive. As Internet technology continues to evolve rapidly and expand its user base, threats and challenges to web application security will also mount. The implementation of a comprehensive web application security program is critical to the stability and trustworthiness of any organization's online assets.

As Internet technology continues to evolve rapidly, threats and challenges to web application security will also **mount.**

Web Applications as the New Security Perimeter

In today's global economy, virtually every enterprise conducts some manner of business online. The applications that drive Internet transactions intersect with multiple technologies and platforms, exposing them to greater security risks. No company is immune to attack. A report released by Gemalto, a leading digital security company, revealed that 945 data breaches led to a massive 4.5 billion data records being compromised worldwide in the first half of 2018. More than 25 million records were compromised or exposed every day, or 291 records every second.[1]

Organizations that develop web applications are under pressure to mitigate these new threats. However, since threats to web applications are constantly changing, so too must the standards and rules for security. Some companies have learned the hard way that a reactive approach to web application security is both costly and inefficient.

For example, testing the security of web applications in post-development can lead to deployment delays and budget overruns. Historically, there has been no economical way to continuously test source code for security issues during the development process. As a result, organizations have been forced to consider security near the end of the development process, when the web application is nearly complete. Typically, web applications are then tested by using tools that require a high level of security expertise or hiring expensive consultants—neither of which is cost-effective.

Clearly, a more proactive approach to web application security is needed. Such an approach should incorporate a well-designed, secure development life cycle and the appropriate tools, processes, and training that will help mitigate risks posed by potential breaches. When developers can address security at the time the first line of code is written, risk and development costs will be lowered in the long run.

The Role of Security During the Software Development Life Cycle (SDLC)

Web application development is an ever-changing industry, and its best practices continue to be defined. Currently, most organizations follow a set of standard steps that outlines each phase of software creation. These phases are collectively referred to as the software development life cycle or SDLC.

The lack of security in web application development has long been recognized as a serious issue and vital missing piece of the process. In the past, security assurance was relegated to the QA phase of development—if it was addressed at all. However, forward-thinking organizations are now adding security activities to every phase of the SDLC, allowing them to discover flaws sooner and significantly increase the security of any applications. The typical security activities in each phase of the SDLC are below:

Training: Everyone involved in web application development is provided basic security training. Scalability and repeatability are critical aspects of effective security training programs.

Requirements: As software requirements are defined, the corresponding security requirements should also be defined. For example, if sensitive customer data is to be collected and stored, there should be established requirements for how the data should be encrypted, both in transit and at rest.

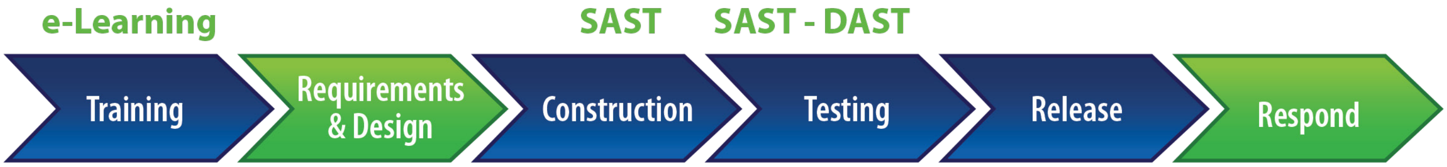
Design: Once the application requirements are captured, architecture is designed to incorporate the software requirements. At this stage of development, the necessary security controls should also be identified and included as part of the application.

Implementation: After requirements have been determined and an architectural design is established, software development begins. Ideally, developers should receive security feedback while they are coding. This feedback must begin as early and as often as possible. Because this phase is often the most labor-intensive, automated security assessments should be conducted continuously, enabling a developer to address issues in near real time.

Quality Assurance: New application code should be tested before it goes into a production environment to ensure that it behaves as expected. While some organizations only test applications to ensure that the functional requirements are met, others are also beginning to test whether the application adheres to established security requirements.

Web application development best practices dictate the importance of **analyzing** source code rather than binary code.

Production: In the deployment phase, continual testing is vital for maintaining security assurance and protecting against common application vulnerabilities. In addition, updates to applications already in production can introduce new flaws. Therefore, all code updates should be subjected to source, QA, and production testing.



The above figure depicts the example of a Secure Waterfall SDLC.

Source Code vs. Binary Code

Web application development best practices dictate the importance of analyzing source code rather than binary code. Source code can be analyzed by the developers who have written it, making it easier for them to identify and mitigate vulnerabilities. In addition, source code review facilitates the analysis of the identified “vulnerable method” calls being made, so that the remediation advice is both pertinent and actionable.

Conversely, binary code analysis is limited because assessments are conducted on compiled code that may not be decipherable by developers. It may be difficult to differentiate code written by internal developers and code inherent to the platform. This shortcoming results in non-actionable and confusing reports for developers. Although source code is preferred, it isn’t always available for analysis. When this happens, the analysis of binary code is a workable option.

Advantages of Static and Dynamic Security Testing

The mitigation of web application security risks is not a one-time exercise. Rather, it is a strategic initiative that requires paying close attention to the changing landscape of emerging risks and deploying new security measures to manage those threats. To do this, an organization must create a security process that tracks an application throughout the SDLC using Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST).

The advantages of leveraging these two testing protocols are as follows:

Static Application Security Testing (SAST)

- Ensures developers can code securely throughout the development process.
- Enables vulnerabilities, including SQL Injection, Command Injection, Improper and/or Custom Cryptography, and a host of other issues that occur primarily on the server to be easily found and eliminated early in the process.

Dynamic Application Security Testing (DAST)

- Makes vulnerabilities like cross-site request forgery, business logic flaws, and some forms of cross-site scripting—which can only be seen once code is live on the web—easier to identify.
- Enables organizations to automatically mitigate identified vulnerabilities using a web application firewall, while waiting for developers to remediate the security risk.

The mitigation of web application security risks is not a one-time **exercise.**

By using SAST and DAST technologies at the appropriate stage of the SDLC, organizations can create a strategic enterprise application security program—but there's more that can be done to shore up web application security. Gartner Research predicts that next-generation modern web and mobile applications will require a combination of SAST and DAST techniques in addition to an interactive application security testing (IAST) approach to be the most effective.[2]

Because IAST combines SAST and DAST techniques, the results are highly actionable and can be linked to a specific line of code, as well as recorded for later replay for developers. Multiple DAST solutions now provide IAST capabilities. However, most IAST solutions also require that an agent be deployed on the application platform, which relegates the technique largely to QA and requires a vendor to explicitly support the platform or language being coded (such as PHP, Java, or .NET/ASP).

SDLC Protects Web Applications from Vulnerabilities

The challenge for many teams is creating a secure software development life cycle. The key to preventing, detecting, and resolving security vulnerabilities during the development life cycle is building a good security architecture based on the following concepts:

Centralize as much as possible: Experienced security engineers have a number of "secrets" for writing secure code, such as centralizing input validation, using a common encoding library, and developing or implementing a centralized authentication and authorization mechanism. Integrating these functions creates a burden of consistency on developers, and consistency supports repeatability. It also requires that the team incorporate concepts, such as authorization schemes and encryption libraries, into the architecture and design.

Educate product development and QA teams: Educating the product team (including the product owner, developer, and tester) incorporates security into every step of the process. Product team members should be trained on encryption, as well as on how to design a good user password reset tool. It's also important to train developers on encoding output in Enterprise Security API (ESAPI) tools. QA engineers need training on how to proxy a web application and perform basic penetration tests. The team's familiarity with security concepts will make security architecture and implementation better.

Detect implementation flaws early with security reviews, static code analysis, threat modeling, and inline penetration testing: By engaging, testing, and evaluating security throughout the life cycle, developers can detect implementation flaws sooner rather than later. Static code analysis is a quick and effective way to scan an entire code-base and detect implementation issues. Threat models enable developers to consider the application as a complete system and identify existing threats. Inline penetration tests, or pen tests, are small, short-cycle analyses that validate security decisions made during feature security reviews. Pen tests also ensure that feature implementations were conducted properly.

Prioritize security during the development life cycle: Too often, quality and security are sacrificed to meet early release dates and push additional features. A team that approaches the implementation of security as a core value can build reliable and secure applications with very little overhead. The buy-in of senior leadership can help reinforce best practices for web application development even under the stress of release dates and client commitments.

Conclusion

Secure web application development is on the brink of revolution. By giving developers continuous feedback during the entire development process and tracking vulnerabilities as they are introduced and remediated, forward-thinking companies will make major headway in developing secure web applications. Addressing security at the beginning of the development process is critical to curbing the current global epidemic of website hacking.

References

1. https://www.gemalto.com/press/pages/data-breaches-compromised-4-5-billion-records-in-first-half-of-2018.aspx
2. https://www.gartner.com/doc/3868966/magic-quadrant-application-security-testing

ABOUT SDG

With more than 30 years of experience partnering with global enterprises on complex business and IT initiatives, SDG is a trusted provider of advisory, transformation, and managed services. The firm empowers organizations to strengthen cyber resilience by integrating AI into identity, threat, and risk management solutions that protect digital assets and deliver measurable business value.
Learn more at www.sdgc.com.

Contact Us: solutions@sdgc.com

SDG

[technology + passion] – risk

- 55 North Water Street
Norwalk, CT 06854
- 203.866.8886
- sdgc.com