Everything About the Secure Software Development Process

31.12.2022

Ismail Tasdelen

Security Researcher

Contents

Everything About the Secure Software Development Process

In this post, I'll be talking about some things about writing secure code. Let's start.

Writing secure code is a crucial skill for any developer. There are several key principles that can help you write more secure code.

1. Keep security in mind throughout the development process. Don't just consider security at the end of the project when it's too late to make significant changes.
2. Use a secure coding methodology, such as the OWASP Top 10, which provides a list of the most common web application security vulnerabilities and recommends strategies for mitigating them.
3. Follow best practices for secure coding, such as properly validating input, escaping output, and using secure cryptographic functions.
4. Avoid common security pitfalls, such as SQL injection, cross-site scripting (XSS), and insecure direct object references.
5. Use a web application firewall (WAF) to protect against common attacks, such as SQL injection and cross-site scripting.
6. Regularly test your code for security vulnerabilities using tools such as static code analysis and penetration testing.

By following these principles and best practices, you can help ensure that your code is secure and less likely to be exploited by attackers.

What are the secure code development processes?

There are several key processes that can help developers write more secure code. These include:

1. **Threat modeling:** This is the process of identifying potential security threats and vulnerabilities in a system and determining the best ways to mitigate them.
2. **Secure coding standards:** These are guidelines that outline best practices for writing secure code.
3. **Code review:** This is the process of having other developers review your code to identify potential security vulnerabilities.
4. **Testing:** This involves using tools such as static code analysis and penetration testing to test your code for security vulnerabilities.
5. **Incident response:** This is the process of responding to and managing security incidents, such as data breaches or attacks on your systems.

By following these processes, developers can help ensure that their code is secure and less likely to be exploited by attackers.

How many steps do secure code development processes take?

There is no specific number of steps that all secure code development processes must follow. The specific steps involved in a secure code development process can vary depending on the organization, the type of software being developed, and the specific security requirements. In general, however, a secure code development process typically includes the following steps:

1. **Identify and prioritize security requirements:** The first step in developing secure code is to identify the specific security requirements that the software must meet. This might include requirements related to data confidentiality, data integrity, access control, and other security concerns.
2. **Design and implement secure code:** Once the security requirements have been identified, the next step is to design and implement the code in a way that meets these requirements. This might involve using secure coding

practices, such as input validation and error handling, to prevent common security vulnerabilities.
3. **Test and validate the code:** After the code has been implemented, it should be tested and validated to ensure that it meets the security requirements. This might involve conducting penetration testing, vulnerability assessments, and other types of security testing.
4. **Monitor and maintain the code:** Once the code has been developed and tested, it should be monitored and maintained to ensure that it continues to meet the security requirements over time. This might involve conducting regular security audits, applying security patches and updates, and responding to any new security threats that arise.

Overall, the number of steps involved in a secure code development process can vary, but these four steps are typically included in most secure code development processes.

What are web application security testing standards?

Web application security testing standards are guidelines and best practices for testing the security of web applications. Some of the key web application security testing standards include:

1. **OWASP Testing Guide:** This guide provides a comprehensive set of procedures for testing web application security.
2. **NIST SP 800–115:** This document provides technical guidance on how to conduct penetration testing.
3. **ISO/IEC 29147:** This standard specifies the requirements for vulnerability disclosure, which is the process of responsibly reporting security vulnerabilities to an organization.
4. **ISO/IEC 27035:** This standard specifies the requirements for incident management, which is the process of responding to and managing security incidents.

By familiarizing yourself with these web application security testing standards and incorporating them into your development process, you can help ensure that your web applications are secure.

What is web application security testing methodology ?

Web application security testing methodology is a set of guidelines and best practices for testing the security of web applications. This methodology typically includes a range of different techniques, such as static code analysis, dynamic testing, and penetration testing.

The goal of web application security testing methodology is to identify and address potential security vulnerabilities in web applications before they are exploited by attackers. By following a thorough and systematic testing methodology, developers can help ensure that their web applications are secure and less likely to be compromised.

Some common web application security testing methodologies include the OWASP Testing Guide, the NIST Cybersecurity Framework, and the Open Source Security Testing Methodology Manual (OSSTMM). These methodologies provide detailed guidance on how to conduct security testing in a way that is thorough, effective, and compliant with industry best practices.

What is DAST?

DAST, or Dynamic Application Security Testing, is a type of security testing that is performed on a web application while it is running. DAST involves sending a variety of inputs to the application and observing its behavior to identify potential security vulnerabilities.

DAST is typically performed by a security tester who has access to the application's user interface and can interact with it in the same way that a real user would. This allows the tester to identify vulnerabilities that may not be apparent from static analysis or other types of testing.

Some common techniques used in DAST include injection attacks, cross-site scripting (XSS) attacks, and authentication bypass attempts. By identifying and addressing these vulnerabilities, developers can help ensure that their web applications are secure.

What is SAST?

SAST, or Static Application Security Testing, is a type of security testing that is performed on a web application's source code. SAST involves analyzing the source code to identify potential security vulnerabilities without actually running the application.

SAST is typically performed by a security tester or a specialized tool that is able to understand the application's code and look for common security issues. Some common techniques used in SAST include scanning the code for known vulnerabilities, looking for insecure coding practices, and checking for insecure configuration settings.

By identifying and addressing these vulnerabilities early in the development process, developers can help ensure that their web applications are secure.

What are the differences between DAST and SAST?

DAST and SAST are both methods of security testing, but they differ in several key ways:

1. DAST is performed on a running web application, while SAST is performed on the application's source code.
2. DAST involves interacting with the application as a real user would, while SAST does not involve actually running the application.

3. DAST is typically performed by a security tester who has access to the application's user interface, while SAST is typically performed by a security tester or a specialized tool that can analyze the source code.

4. DAST focuses on identifying vulnerabilities that are present in the running application, while SAST focuses on identifying vulnerabilities that may be present in the source code.

Overall, DAST and SAST are complementary techniques that can be used together to provide a more comprehensive view of an application's security.

What is OSSTMM?

OSSTMM, or the Open Source Security Testing Methodology Manual, is a set of guidelines and best practices for conducting security testing. It covers a wide range of topics, including how to scope and plan a security test, how to conduct the test itself, and how to report on the results.

The OSSTMM is designed to be a comprehensive and flexible framework that can be used by security testers of all skill levels. It is particularly useful for testing the security of web applications, networks, and other types of systems.

One of the key features of the OSSTMM is that it is open source, which means that it is freely available and can be modified and adapted to meet the specific needs of an organization or project. By using the OSSTMM, developers can ensure that their security testing is thorough and effective.

What tools can be used in source code analysis?

There are several tools that can be used in source code analysis, also known as static application security testing (SAST). These tools are designed to help developers identify potential security vulnerabilities in their source code before the application is deployed.

Some common tools used in source code analysis include:

1. **Static code analysis tools:** These tools scan the source code for known vulnerabilities and insecure coding practices. Examples include Fortify, Veracode, Checkmarx, and SonarQube.
2. **Compiler-based tools:** These tools use the compiler to analyze the source code for potential security vulnerabilities. Examples include Microsoft Security Code Analysis and Coverity.
3. **Interactive development environment (IDE) plugins:** These are plugins that can be installed in an IDE, such as Eclipse or Visual Studio, to provide security analysis as the developer is writing code. Examples include FindBugs and PMD.

By using these tools, developers can identify potential security vulnerabilities in their source code and take steps to address them before the application is deployed.

What tools can be used in web application security testing?

There are several tools that can be used in web application security testing. These tools are designed to help identify potential security vulnerabilities in web applications before they are deployed.

Some common tools used in web application security testing include:

Dynamic application security testing (DAST) tools: These tools simulate an attack on the web application and monitor its behavior to identify potential vulnerabilities. Examples include Burp Suite and Acunetix, Netsparker and IBM AppScan Enterprise.

What are the popular Burp Suite add-ons?

Burp Suite is a popular tool used for web application security testing. It is a modular platform that allows users to extend its capabilities through the use of add-ons, also known as extensions. Some of the most popular Burp Suite add-ons include:

1. **ActiveScan++:** This add-on improves the capabilities of Burp Suite's built-in active scanning feature, allowing it to find more vulnerabilities and provide more detailed information about each one.
2. **J2EEScan:** This add-on is designed to automate the testing of Java EE applications, which are often used in enterprise environments. It can help identify a wide range of vulnerabilities, such as insecure configuration settings and dangerous method calls.
3. **Param Miner:** This add-on is designed to help testers identify hidden parameters in web applications, which can be a source of vulnerabilities. It does this by automatically sending requests with various parameter values and analyzing the responses to identify hidden parameters.
4. **WSDLFuzz:** This add-on is designed to automate the testing of web services, which are often used to provide APIs for web applications. It can help identify a wide range of vulnerabilities, such as XML injection and insecure object deserialization.

By using these and other add-ons, users of Burp Suite can extend its capabilities and make it even more effective at identifying security vulnerabilities in web applications.

What are free web application security tools?

There are several free web application security tools that are available for developers to use. These tools can help identify potential security vulnerabilities in web applications and provide recommendations for addressing them. Some examples of free web application security tools include:

1. **Burp Suite Community :** The Burp Suite community is a group of users who are interested in web security and use the Burp Suite software to test the security of their own websites. The community includes security professionals, developers, and enthusiasts who share their knowledge and expertise on using Burp Suite to find and fix security vulnerabilities in web applications. Burp Suite is a popular tool used by security experts to perform security testing and penetration testing on websites and web applications. The community provides a platform for users to ask questions, share tips and tricks, and discuss the latest developments in web security.
2. **OWASP ZAP:** This is an open-source web application security testing tool that can be used to identify a wide range of vulnerabilities, such as SQL injection and cross-site scripting (XSS). It is available for free and can be run on Windows, Linux, and macOS.
3. **Brakeman:** This is an open-source security scanner for Ruby on Rails applications. It can help identify a wide range of vulnerabilities, such as cross-site scripting (XSS) and insecure deserialization. It is available for free and can be run on Windows, Linux, and macOS.
4. **Arachni:** This is an open-source web application security scanner that can be used to identify a wide range of vulnerabilities, such as SQL injection and cross-site scripting (XSS). It is available for free and can be run on Windows, Linux, and macOS.
5. **Ratproxy:** This is an open-source web application security testing tool that can be used to identify potential security vulnerabilities in web applications. It is available for free and can be run on Linux, macOS, and other Unix-like operating systems.

How should a penetration test process be?

A penetration test, or pentest, is a type of security testing that involves simulating an attack on a computer system or network to identify vulnerabilities. The process of conducting a pentest should follow a few key steps to ensure that the test is thorough, effective, and compliant with industry best practices.

1. **Scope and plan the test:** Before starting the pentest, it is important to define the scope and goals of the test, as well as the specific systems and networks that will be tested. This will help ensure that the pentest is focused and targeted, and that the results are meaningful and useful.
2. **Conduct the test:** Once the scope and plan have been defined, the actual pentest can be conducted. This typically involves using a variety of tools and techniques to simulate an attack on the target systems and networks.
3. **Analyze the results:** After the pentest has been completed, the results should be analyzed to identify any vulnerabilities that were discovered. This typically involves reviewing logs, reports, and other data to determine the types and severity of the vulnerabilities that were identified.
4. **Report the findings:** Finally, the results of the pentest should be documented in a comprehensive report that includes details about the vulnerabilities that were discovered, as well as recommendations for addressing them. This report should be shared with the relevant stakeholders to ensure that the vulnerabilities are addressed and the systems are made more secure.

By following this process, organizations can conduct effective and thorough pentests that help identify and mitigate security vulnerabilities.

What are the professions in web application security?

There are several professions in web application security that involve working to identify and address security vulnerabilities in web applications. Some examples of these professions include:

1. **Web Application Security Engineer:** This is a professional who is responsible for designing, implementing, and maintaining the security of web applications. This may involve developing secure coding practices, conducting security testing, and managing security incidents.
2. **Web Application Security Tester:** This is a professional who is responsible for testing web applications for security vulnerabilities. This may involve using tools such as dynamic application security testing (DAST) and static application security testing (SAST) to identify vulnerabilities and recommend solutions.
3. **Security Consultant:** This is a professional who provides expert advice and guidance on web application security. This may involve conducting security assessments, providing training on secure coding practices, and helping organizations develop and implement security policies.

By pursuing one of these professions, individuals can help ensure that web applications are secure and less likely to be exploited by attackers.

What will the cyber security professionals look like in the future?

It is difficult to predict exactly what cyber security professionals will look like in the future, as the field is constantly evolving and new technologies and threats are emerging all the time. However, it is likely that cyber security professionals will need to have a broad and deep understanding of both technology and business in order to be successful.

In the future, cyber security professionals may need to have expertise in a wide range of areas, such as cloud computing, artificial intelligence, and the Internet of Things (IoT). They may also need to have a deep understanding of the business implications of security and be able to communicate effectively with a variety of stakeholders, including executives, IT staff, and customers.

In addition, it is likely that cyber security professionals will need to be lifelong learners who are able to keep up with the latest developments in the field and adapt to new technologies and threats as they emerge. By staying current and continuing to learn and grow their skills, cyber security professionals will be well-positioned to succeed in the future.

What is a bug bounty?

A bug bounty is a reward offered by an organization to individuals who find and report security vulnerabilities in their systems or applications. These rewards, also known as "bounties," are typically offered as part of a bug bounty program, which is a voluntary initiative that encourages people to identify and report security issues. The goal of a bug bounty program is to identify and fix vulnerabilities in a system or application before they can be exploited by malicious actors. Organizations that offer bug bounties typically have a set of guidelines and rules that determine which types of vulnerabilities are eligible for a reward and how much the reward will be. Bug bounty programs are becoming increasingly popular as a way for organizations to improve the security of their systems and protect against cyber attacks.

How to protect from web application security vulnerabilities?

There are several steps that developers can take to protect their web applications from security vulnerabilities. These include:

1. **Follow secure coding practices:** This involves following best practices for writing secure code, such as properly

validating input, escaping output, and using secure cryptographic functions.

2. **Use a web application firewall (WAF):** A WAF is a tool that can be installed in front of a web application to provide protection against common attacks, such as SQL injection and cross-site scripting (XSS).
3. **Conduct regular security testing:** This involves using tools such as static code analysis and penetration testing to test your web application for security vulnerabilities on a regular basis.
4. **Keep your software and libraries up to date:** This involves regularly updating your web application and the libraries and frameworks it uses to the latest versions. This can help ensure that you are using the most secure versions of the software and are not vulnerable to known security vulnerabilities.

By following these steps, developers can help protect their web applications from security vulnerabilities and keep their systems secure.

In general, I answered the question of how you can improve yourself in secure software development process. Take care and see you in my next article.

"People trust easily, as I discovered at a very early age."
Kevin D Mitnick