10 BEST PRACTICES FOR SOFTWARE DEVELOPMENT SECURITY

Nov 29, 2021

Øyvind Forsbak
CTO

TYPES OF SOFTWARE ENGINEERING PRACTICES

Software security is a critical topic that should be given due attention. Software development has become an integral part of our lives, and we rely on it for almost everything we do.

It is important for developers should follow best practices for software development security. The purpose of these best practices is to minimize any vulnerabilities in your code, protect it from hackers and cybercriminals, and maintain users' privacy.

This blog post will list down some best practices that you should follow while developing your software applications.

WHAT ARE THE MOST COMMON SECURITY RISKS?

Before listing the best practices, it is vital to understand the most common security risks developers face. Some of the common security risks faced by software developers include:

Software systems not actively being maintained

If you are not developing your application anymore or being supported by a small team, there are high chances that the software applications have vulnerabilities. Once these vulnerabilities get exploited, the hackers could access

secure data and confidential information stored on your server leading to various security issues.

Poorly written code

Another common risk in software development is poorly written code. An application with poorly written code is challenging to secure, and coding practices such as input validation, output encoding, error handling, secure storage, and secure coding practices are often not followed.

Vulnerable web services

Web services often store sensitive data related to the user and personal information. If the web services have vulnerabilities, hackers could exploit them to access sensitive information or perform unauthorized activities on your website.

Insecure password storage

Passwords are often stored in a way that makes it easy for attackers to steal and decrypt passwords using various techniques such as dictionary attacks and brute force attacks. Use strong cryptography to secure your passwords.

Legacy software

Legacy software is vulnerable to security attacks. They are usually written without secure coding practices and are not updated frequently, making them prone to cyber-attacks & data breaches.

WHY DO DEVELOPERS SKIP SECURITY PREPARATIONS?

While secure software development is a necessary process, it is often skipped by developers due to various reasons.

The most common reason is time and resource constraints. Developers often find themselves in a dilemma where they have too much work on their plate and not enough time or resources for everything that needs to be done before the release date. As a result, they end up taking shortcuts by focusing only on what's required at the moment.

Another reason is lack of awareness about potential threats. Many programmers believe that hackers will never attack their application, so there's no need for this additional step during the development phase, which can eat into precious person-hours

TOP 10 SECURITY PRACTICES FOR SOFTWARE DEVELOPMENT

Let's go through some best practices that should be included in secure software development:

1. Treat Software Security as a Priority Right From The Start

Security should be considered from the planning stages of your project. Security starts with requirements, so it is essential to think about what vulnerabilities may come up in each stage of software development. This means that security should always be evaluated when making changes or adding features later on down the line.

Secure software development lifecycle (SDLC) is a way to develop secure applications. It takes into account the security risks involved throughout the entire application lifecycle. Furthermore, it works through each phase to ensure that appropriate controls are implemented at every process step.

2. Conduct Security Awareness Training

Your software developers need to know what they are up against. They should be informed of common attacks in the software development world and how to avoid them.

Security awareness training should include information about common software development vulnerabilities. It should also include information about how hackers and cybercriminals work.

Developers need to know what mistakes they are likely to make when writing code to avoid making those same mistakes themselves. Education and knowledge transfer will help your software developers write secure applications from day one.

As part of security awareness training, it's good to hold regular meetings where everyone gets together and discusses secure development practices. These meetings can be very beneficial when it comes to how to identify vulnerabilities with your code before cyber-attackers do!

3. Use Code Reviews to Identify Potential Security Threats

Code reviews help developers identify and fix security vulnerabilities so they can avoid common pitfalls. Secure design is an integral part of software development. When writing code, adopt a defensive mindset that helps you write as little code as possible. You should also be testing your code and writing unit tests for all areas of concern.

For every code change you make, you should go back and check to see if those changes have introduced any new security vulnerabilities. In addition, it is essential to review security requirements to ensure that secure coding practices are followed throughout the development process.

4. Use Static Code Analysis Tools

Security mistakes can sometimes seem subtle and can be easily overlooked by experienced developers. Static code analysis tools can bridge this understanding gap, find security vulnerabilities, and facilitate code review processes.

Before your software is deployed, static code analysis tools are an excellent approach to finding software vulnerabilities. It can be integrated into the pipeline so that every time there is a new build, and it will automatically run through these checks and flag any potential issues.

During a security code review, static code analysis tools may be used to identify areas of concern. These tools are essential for large organizations where developers may come and go or lack security knowledge.

Static code analysis tools are not perfect, but they can certainly help catch some of the most common issues that lead to software vulnerabilities such as SQL Injection, Cross-Site Scripting (XSS), and sensitive data exposure.

5. Use Popular and Well-Maintained Libraries and Frameworks

It is best to use popular, well-maintained libraries or frameworks when writing software since they are less likely to have vulnerabilities than newly created code bases.

Using open source components can help you better manage your software security since you can benefit from early bug detection and patches. In addition, using secure software development libraries can help reduce your application's attack surface and make it more secure.

Developers should always research the reputation of a library or framework before using it extensively in their applications. They can use online tools that provide detailed information about each project's community activity, release frequency, and other metrics, which will help them make an informed decision on whether this component is secure enough for their needs.

6. Use Popular and Well-Maintained Libraries and Frameworks

Get your team to know OWASP's Top Ten Software Vulnerabilities. These web application security flaws are the most common mistakes that secure software development best practices avoid.

Having an updated list of the top software vulnerabilities in one place makes it easy for developers to ensure that they are taking steps to avoid these common errors. Even if your team is not familiar with OWASP, knowing their most important points will help you decide how best to secure your software development projects.

7. Secure coding guidelines and standards

Secure software development starts with coding guidelines and standards. Your organization's secure coding guidelines and measures should be defined by a consensus of experts, taking into account industry best practices.

Secure coding standards can help promote better design principles within an organization, which reduces vulnerabilities before software goes live. Moreover, by providing a standard set of rules and restrictions regarding what kind of code gets written, teams can enforce reliable testing methods throughout the Software development lifecycle to ensure that they are not introducing new vulnerabilities.

Threat modeling is another technique that software developers should use to secure software. Threat modeling identifies threats by looking at specific data flows and then analyzing what can go wrong within each flow.

Here are some concepts that developers should know to create your secure coding guidelines:

Encryption

All data should be encrypted in transit and at rest. This includes database storage, file storage, sessions, cookies, etc. Encryption is the only way to maintain the confidentiality of user data on a network where any node can potentially become compromised by an attacker who will have full access to all traffic they observe (i.e., plain text).

Password Hashing

To secure passwords, never store them in plain text. Instead, use a password hashing algorithm to compute a unique hash of the user's password that can be stored in your database.

SQL Injection

SQL injection attack is when a hacker inserts a SQL query through an application interface to extract or manipulate data from the back-end database. SQL injection attacks can be prevented by using parameterized queries instead of dynamic SQL statements.

Cross-Site Scripting (XSS)

XSS is a type of attack that occurs when an attacker injects malicious scripts into the application. This kind of attack aims to get users to click on links that will then send them to malicious sites or have software deliver malware directly onto their devices without any action required by the user.

Sensitive Data Exposure

Data Exposure occurs when encryption keys, passwords, Social Security numbers, credit card information, and other personally identifiable details are not adequately protected from hackers. Sensitive data should be encrypted both in storage and when transmitted over the Internet. The sensitivity of particular pieces of information varies, but there are tried-and-true ways to determine what sensitive data needs to be protected by default.

Input Validation Attacks

Input validation attacks are when an attacker finds a way to manipulate the application into accepting data that it shouldn't. This is not strictly limited to SQL injection but can include input from outside sources such as network packets or user-generated content like text messages and email address identifiers.

Buffer Overflow Attack

These attacks exploit the fact that when an application allocates space for input data, it can access memory beyond its given boundaries. As a result, hackers introduce more code into a program's buffer than developers anticipated during the software development process and then execute this excess data to gain control of the app or system.

Unvalidated Redirects & Forwards

This is where attackers can redirect users from legitimate websites onto malicious ones without warning them about the switch beforehand. In addition, by using unauthenticated parameters within requests, hackers can often get away with changing which page is being displayed.

Improper Error Handling

Improper error handling is when an application fails to provide developers with a way of handling unexpected errors. This can allow hackers to execute their code or gain access through back-end servers by exploiting error messages that are not handled accordingly.

Application Whitelisting (aka: “Least Privilege”)

The concept of least privilege is where applications are given access to only the minimum resources needed to run securely. This way, if there’s a vulnerability in one of your web apps or back-end services, it can’t be used as an entry point by hackers looking for exploitable weak points.

Insufficient Logging & Monitoring

Often software has insufficient logging and monitoring capabilities which can make it difficult (if not impossible) for developers to determine if an attack has taken place.

Attackers usually don’t want their actions logged so they can stay undetected. Therefore, developers should implement proper security monitoring and auditing practices, including user activity tracking, file integrity monitoring, and network activity logs.

8. ISO 27001 Certification

You should also consider getting your company ISO 27001 certified. ISO 27001 is a worldwide information security standard that outlines security standards for developing, implementing, maintaining, and improving an Information Security Management System.

ISO 27001 certification can help secure software development by increasing an organization’s ability to protect confidentiality, integrity, and availability of critical business information.

9. Penetration Testing

Penetration testing is an automated way of identifying potential security issues in your software. Proper penetration tests can be done by hiring a penetration testing team that specializes in software security.

These security experts use the same tools as hackers to evaluate how secure your system is against these types of attacks. In most cases, companies should do some form of penetration testing every month on a subset of their systems or products. This way, you can have confidence that any existing vulnerabilities are quickly being addressed and resolved before attackers find them first.

Security testing also involves identifying and mitigating concerns around third-party software components. In addition, companies should secure their code and ensure that vendors' and partners' products are secure as well.

10. Incorporate secure software development practices into your DevOps practices

Once you have started implementing these best practices, make sure to integrate them into your DevOps processes. This will allow the entire software development team to be aware of security requirements and build secure software.

As a result, your team can identify security issues at the beginning of development instead of waiting until it's too late. This is why secure DevOps (or DevSecOps) practices are so important when dealing with secure software development from start to finish to reduce vulnerabilities and eliminate bugs before they impact end-users.

If necessary, companies can even implement a bug bounty program with rewards for identifying security bugs in their apps or services. Finally, it's important to regularly communicate progress updates within your company so that people understand where all these new policies are coming from and why they're needed.

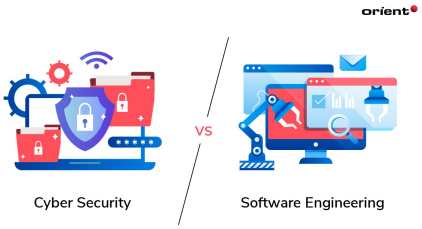
CONCLUSION

In conclusion, secure software development is about more than just secure code. It's essential to take a holistic approach and implement certain DevOps practices into your everyday workflow. When we say secure DevOps, we mean it: from the beginning of Software Development through deployment and beyond. This ensures that security becomes an integral part of everything you do - not something on its own that only gets attention at specific intervals or when there's been a breach.

In the end, secure software development is a journey that never ends. Therefore, you should always look for new ways to improve and make your code more secure as technology evolves and hackers find new types of attacks to exploit against Software vulnerabilities.

Topics: Software Security

YOU MIGHT ALSO LIKE



[What is the Difference Between Cyber Security vs Software Engineering?]

[Why You Should Implement Security as Code Into Your DevOps Project]

[Should or Should Not Outsource a Cyber Security Team?]

LOOKING FOR AN IT PARTNER?

Contact us today for a free quote within 3 business days

FOLLOW US

Your name*

Your email*

Company name*

I'm interested in

Your message...*

I'm not a robot

reCAPTCHA
Privacy - Terms

SEND

Company

Our Services

Our Expertise

Advanced Technologies

Resources

Contact Us

Head office - Ho Chi Minh City