Table of Contents

Subscribe for weekly updates

Your email

I agree with CyStack's Terms of Service and Privacy Policy

Subscibe to Newsletter

Share

OPERATIONS SECURITY

Top Risks In Software Development Life Cycle: 7-Minute Read

Jenny Duong

Marketing Executive | April 5, 2023

Security Risks in SDLC

Let's discuss **Security risks in Software Development Life Cycle (SDLC)**.

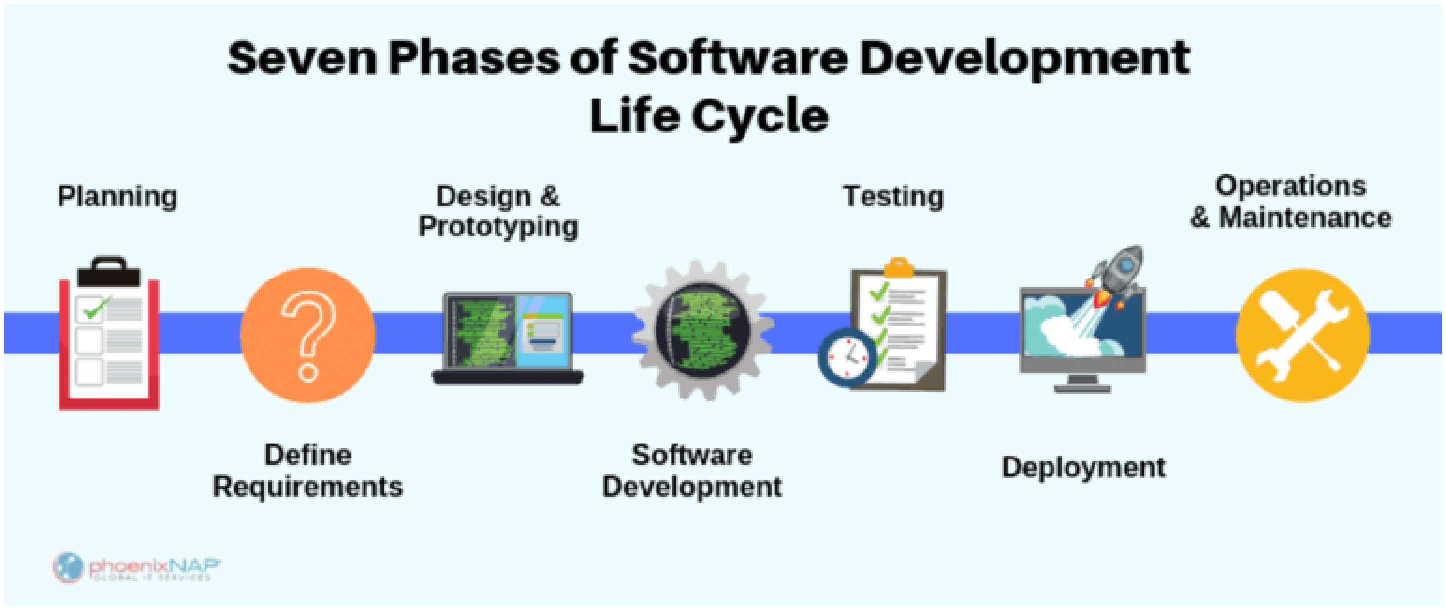
Gone are the days when security assessment was only implemented during the testing phase. Nowadays, cybercriminals and hackers are trying to exploit vulnerabilities at any time of development and you always need to keep your guard up. Specifically, implement continuous security testing throughout the project to troubleshoot underlying security risks and fix them early at a minimum cost.

This article provides useful information to do so, with the following topics:

- What Is Software Development Life Cycle (SDLC)?
- Why Is SDLC Security Important?
- Common Security Risks in Software Development Life Cycle (SDLC)
- Best Security Practices in SDLC

Scroll down to read more!

What Is Software Development Life Cycle?



Seven typical phases of Software Development Life Cycle – SDLC (Source: PheonixNAP)

Software Development Life Cycle (SDLC) is an effective structured procedure to create quality and low-cost software in an optimal timeline. Software developers must accordingly outline what to do in each stage/phase and predict risks before starting.

By doing so, the SDLC reduces unnecessary costs and efforts throughout its cycle of seven phases.

Planning

Two central questions of this phase are:

- **What do we want?**

The development team tries to collect as much input and feedback from stakeholders getting involved and benefiting from the project. Afterward, the team determines software development's feasibility, then defines the scope and creates a timetable with objectives and target goals. We suggest the team plot potential risks and backup plans for risk mitigation or elimination.

- **What do we need to develop the software?**

Next, software developers continue to determine the resources and costs required to implement the software effectively. The typical outcomes are capacity planning, resource allocation, cost estimation, projected schedules, leadership structure, risk management framework, and provisioning.

Elicit and Define Requirements

In some software development life cycles, defining requirements is included in the planning phase. It is also involved in coll[illegible]ng requirements from various stakeholders and determining resources.

However, the requirements gathered during the planning phase are usually high-level. You can detail specific requirements in the second phase following the planned timetable.

Design/Prepare Prototyping

The designing and/or prototyping phase helps translate elicited requirements into visual forms.

- **Architecture design**: Programing language, overall design, industry practices, templates, or boilerplates.
- **Platforms:** Platforms used to run the software, for instance: Android, Windows, IOS, Linux, etc.
- **User Interface**: Specific ways users can interact with the software and the responses of the software to input.
- **Communications**: Specified communication methods with external assets (server, instances, 3rd integrations.)
- **Security**: Measures implemented to secure the software, such as traffic encryption, SSL, and password protection.

Prototyping is an early design version – helpful in explaining how the software looks and interacts. Feedback, if any, is collected to improve and update the final design.

Develop The Software

Next, write the actual software. Depending on the scope and timeline, the software is assigned to several developers or teams in different tasks.

Tasks might hold up the software development phase if they are under compiling codes or waiting for test results. The SDLC's role is to predict such delays and plan for the developers to do other tasks or duties.

Implement Testing

No software applications are bug-free.

Thus, this phase is essential to ensure functions work as requested in the requirements and designing stages before being available to customers. This helps enhance user experience and satisfaction.

Different tests are done, such as code quality testing, unit testing, system testing, integration testing, security testing, performance testing, and more. We also recommend running automation testing methods to reduce time and human resources.

Deploy The Solution

The software is made usable to its customers through deployment only when passing test cases.

The process can be as simple as downloading the application link on the website or app stores. In some cases, it can be complex as integrating or upgrading the entire company's database to a new application.

Operate and Frequently Maintain

We haven’t finished the software development life cycle yet. Developers should continue operating and maintaining the application.

For example, customers can report bugs not covered during the testing phase. Alternatively, the team themselves detect vulnerabilities to fix and improve. Either way, it might request new development cycles.

Why Is SDLC Security Important?

Why cover security risks in Software Development Life Cycle?

AS As we mentioned, security testing used to be a singular and separate task in the testing phase. Consequently, many bugs and vulnerabilities still exist without notice or late notice. There were cases when the requirements and coding parts were leaked to hackers or cybercriminals.

Software development teams now, however, must prioritize and bake security measures and practices in all phases and particularly the early ones of the SDLC.

With this idea in mind, the industry introduced and recognized a “Secure SDLC” as a significant concept. Even big tech co mpanies have invested significantly in security. The advantages of the approach include but are not limited to:

- Increasing awareness of security among stakeholders and improving trustworthiness.
- E[illegible] detection of flaws in the software and fixing them quickly with a significant cost reduction.
- Enhancing the overall safety of the software and indirectly reducing intrinsic business risks.

Common Security Risks in Software Development Life Cycle (SDLC)

Security risks in software development life cycle phases

Important as the Secure SDLC is, many security risks in Software Development Life Cycle occur without being foreseen due to the complex telecommunications, hacking activities through power systems, underlying internet-ware applications, etc.

It would be helpful if you keep yourself updated on the common risks of each phase and get prepared.

Risks in Planning Phase

Planning is the first stone for software development, in general, and security practices, in particular.

Yet, inadequate security training and awareness is a popularly overlooked risk in this phase. Imagine the supervisors do not appreciate the value of security and spend less on security activities; the employees will also ignore security practices in their tasks and actions.

In addition, there might be no specific framework for risk management stated during the planning. In the subsequent phases, when the risks happen, the development team does not know how to resolve them effectively. The delay in tackling risks results in high costs and a waste of resources.

Risks in Requirements Phase

We are trying to gather as much information as possible, and extensive data are stored or documented for further reference in the development and testing phases.

You had better be careful against external security risks such as viruses, hacking, denial of services or power loss or unsystematic risks such as data loss, human error, or inside attacks.

Another security risk during the requirements phase is the shortage of security requirements which indicates what the software should not do. This is common since many development teams only spend time on functional requirements.

Risks in Designing Phase

As mentioned above, data exposure due to external or unsystematic risks is also among the underlying security issues during the designing phase.

Moreover, development teams usually forget architecture design adaptive to possible environmental changes. As a result, software development faces the risk of insecurities in changing cases.

Risks in Software Development

Experts confirmed that security risks occur most frequently during software development. The main reason is access control and source code management since the project is often divided into parts and assigned to different developers.

The writing codes might not be consistent from a security perspective. Plus, there is not enough attention to access and update history. At that time, the team needs to define well-secure coding guidelines and code-reviews procedure in the first place.

Risks in Testing Phase

Security problems are more complex to test than functions since they lack acceptance criteria, and the abstract properties for security testing are not inherently checkable.

For instance, test cases for crashing situations often involve combined bugs at a time. Then, there is a risk that testers and developers cannot replicate the security problems and investigate the issues.

Risks in Deployment Phase

Deploying an application is a sensitive process due to various environments. During the transfer time, the code might be exposed to hackers trying to find vulnerabilities across systems and steal the data or damage the systems.

Risks in Operation and Maintenance

Even if the software is installed and runs smoothly on the live system, it is still under the possible security risks: both systematic and unsystematic, such as data exposure, hacking efforts, etc.

Best Security Practices in SDLC

Best practices to tackle security risks in the software development life cycle

Although the risks might differ from phase to phase, we have standard security practices for your attention to minimize the chances of security risks in the software development life cycle.

- Educating the development team on security SDLC
- Building risk management strategies to handle risks in SDLC
- Eliciting requirements carefully, including security requirements
- Building secure coding checklists and doing code reviews
- Running automatic test cases to detect difficult risks

Your Takeaway

Take notes of all security risks in the Software Development Life Cycle and discuss them next time in the planning phase of your project – be better than sorry. The earlier you detect the security risks, the more quickly and less money you have to spend during the project!

Related posts

Cybersecurity Framework: Building A Security Model For Businesses

2022 Trends of Cyber Threats: Know to Prevent!

March 24 2023 | OPERATIONS SECURITY

Advanced technologies are like a double-edged sword. While they revolutionize our life and how we do business, technologies also expose us to higher cyber attacks. Thus, it is important to always update yourself on the latest trends of cyber threats and, more importantly, how to prevent or minimize th…

Address: Tan Hong Ha Complex, 317 Truong Chinh Street, Thanh Xuan District, Hanoi, Vietnam
Email: contact@cystack.net
Phone: (+84) 247 109 9656

EN VI

SECURITY ASSESSMENT

Vulnerability Assessment

Internal Network Audit

Managed Bug Bounty

Cloud Security Audit

Performance Test

Penetration Test

OPERATIONS SECURITY

Managed Security Service

Security Monitoring

Incident Response

Security Training

DevSecOps

Vulnerability Management

DATA SECURITY

Data Loss Prevention

Password Manager

BLOCKCHAIN SECURITY

Blockchain Protocol Audit

Smart Contract Audit

On-chain Monitoring

RESOURCES	COMPANY
Blog	About Us
Research	Brand

© 2023 by CyStack.,JSC. All rights reserved.