Secure software development

When it comes to creating, releasing, and maintaining functional software, most organizations have a well-oiled machine in place.

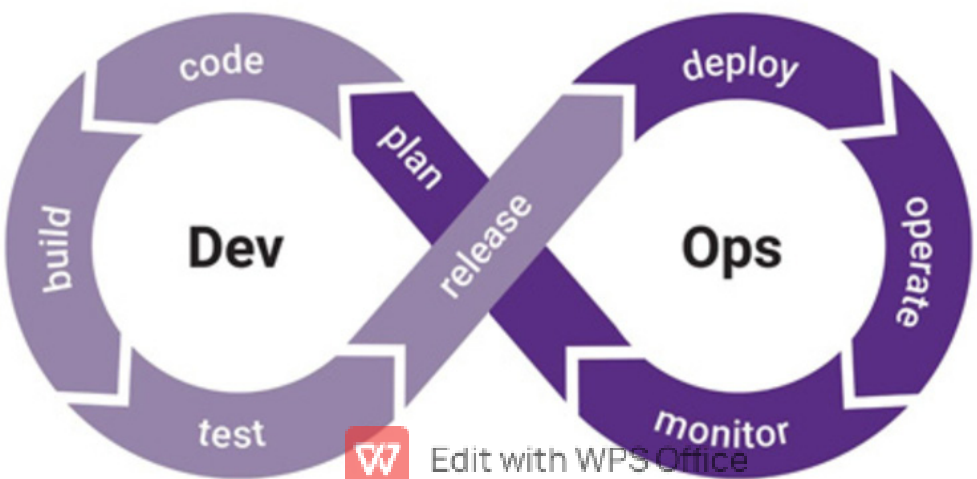
However, when it comes to securing that software, not so much. Many development teams still perceive security as interference—something that throws up hurdles and forces them to do rework, keeping them from getting cool new features to market

Cont

But insecure software puts businesses at increasing risk. Cool new features aren’t going to protect you or your customers if your product offers exploitable vulnerabilities to hackers. Instead, your team needs to integrate security into the entire software development life cycle (SDLC) so that it enables, rather than inhibits, the delivery of high-quality, highly secure products to the market.

What is the secure SDLC and why should I care?

A software development life cycle (SDLC) is a framework for the process of building an application from inception to decommission. Over the years, multiple SDLC models have emerged—from waterfall and iterative to, more recently, agile and CI/CD, which increase the speed and frequency of deployment.



Cont.

In general, SDLCs include the following phases:

- Planning and requirements
- Architecture and design
- Test planning
- Coding
- Testing and results
- Release and maintenance

Cont.

In the past, organizations usually performed security-related activities only as part of testing—at the end of the SDLC. As a result of this late-in-the-game technique, they wouldn't find bugs, flaws, and other vulnerabilities until they were far more expensive and time-consuming to fix. Worse yet, they wouldn't find any security vulnerabilities at all.

The Systems Sciences Institute at IBM reported that it cost six times more to fix a bug found during implementation than one identified during design. Furthermore, according to IBM, the cost to fix bugs found during the testing phase could be 15 times more than the cost of fixing those found during design.

Cont.

So it’s far better, not to mention faster and cheaper, to integrate security testing across the SDLC, not just at the end, to help discover and reduce vulnerabilities early, effectively building security in. Security assurance activities include architecture analysis during design, code review during coding and build, and penetration testing before release.

Advantages of a secure SDLC approach:

- Your software is more secure, as security is a continuous concern.
- All stakeholders are aware of security considerations.
- You detect design flaws early, before they’re coded into existence.
- You reduce your costs, thanks to early detection and resolution of defects.
- You reduce overall intrinsic business risks for your organization.

How does a secure SDLC work?

- Generally speaking, a secure SDLC involves integrating security testing and other activities into an existing development process. Examples include writing security requirements alongside functional requirements and performing an architecture risk analysis during the design phase of the SDLC.
- Many secure SDLC models are in use, but one of the best known is the Microsoft Security Development Lifecycle (MS SDL), which outlines 12 practices organizations can adopt to increase the security of their software

What is Secure software development

Secure software development is a methodology (often associated with DevSecOps) for creating software that incorporates security into every phase of the software development life cycle (SDLC).