CRASHTEST **SECURITY**

CONTINUOUS SECURITY

IMPROVE WEB APPLICATION SECURITY BY USING CONTINUOUS SECURITY SCANS

THE WAY TO SECDEVOPS
CONTINUOUS SECURITY

AGILE SOFTWARE DEVELOPMENT REQUIRES AGILE SECURITY TESTING

CONTINUOUS SECURITY GUARANTEES A SECURE SOFTWARE AFTER EVERY RELEASE.

New kinds of software development also require a new way of testing. The days when a software project took a year from initiation to deployment after a manual security check are over. Manual security testing can no longer keep pace with new software often rolled out several times a week.

Therefore, testing needs to be automated and integrated into developers' daily workflow. The Open Web Application Security Project (OWASP) calls this form of testing continuous security (testing).

Quality management and problems are well researched. For example, the rule of ten states that an error detected at a certain stage of product development costs ten times more money to correct than if this error had been found one stage earlier. Compared to a mistake in the planning phase, an error in the production phase can cost up to 1000 times more. These figures are similar for security errors (i.e., vulnerabilities) in software products.

In agile software development, where a two-week development sprint covers all phases from planning to development to customer presentation, the value of testing increases radically.

Continuous Security guarantees that all software versions are tested during the development phase. Unlike manual security testing, which is often only performed for major releases, all versions are released after testing. This type of testing enables early error detection in the development lifecycle, which leads to time and cost savings while increasing the security level of the developed product.

1. SECURITY CHALLENGES IN AGILE SOFTWARE DEVELOPMENT

Companies face several security challenges when moving to or using agile software development in today's business world. The following chapter outlines the most critical challenges.

- Importance of time-to-market
- Wide range of security tools
- Lack of security expertise
- Sole reliance on frameworks

1.1 FUNCTIONALITY AND TIME-TO-MARKET ARE MORE IMPORTANT THAN SAFETY

One of the main reasons for implementing Continuous Integration / Continuous Deployment is to be more responsive to customer needs and to shorten software development cycles. Usually, the increased pressure to develop more and more features leads to negligence in security testing.

Research shows that security is often not considered a business-critical issue. Security testing is only done at the customer's specific request, or the potential risk in releasing the subsequent significant versions of a piece of software seems to require it. This means that version 1.0 has to go through a (manual) penetration test (pentest), and the next version to be tested for vulnerabilities is often only version 2.0. All minor versions do not go through an additional security test and are released untested.

Essentially, the security level is based solely on the skills of the software developers. This is due to two reasons: 1. security testing and correction of results takes time; 2. security testing, especially manual testing, is costly. To get the software launched on time and within budget, the developers' focus is on creating features, not on increasing the security of the software.

1.2 WIDE RANGE OF SECURITY TOOLS COVERING ONLY INDIVIDUAL VULNERABILITIES

A second challenge for today's software developers is the multitude of different tools available on the market. A specific test for almost every security vulnerability checks that one vulnerability very well but only covers one vulnerability.

For example, SQLMap is a great tool to find SQL injection vulnerabilities, but that is only one of the attack vectors hackers use. If a developer uses multiple of these scanners, they face the next challenge.

Each scanner must be individually configured and customized to meet the needs of the organization. The scanners are based on different programming languages and often come with other requirements such as system libraries, so it takes the developer a lot of time to set up the internal security scanners. WIn addition, whent here are several different security tools, developers must decide which one to use. Since they often do not have specific security training, evaluating the tools is complicated.

In addition, the output formats differ from tool to tool, so a vulnerability management solution also comes into play, which must be configured and set up. If the developer wants to run a security scan, he must then run each scanner individually and consolidate and standardize the results by hand. He must also ensure that each scanner is always up to date to detect the latest vulnerabilities.

1.3 SECURITY EXPERTISE IS NOT EASY TO COME BY FOR SMALLER COMPANIES

Another problem that mainly concerns management is the lack of qualified software developers - especially in security.

In particular, it is challenging for small and medium-sized companies to recruit the experts they need in security management. This is another reason why security testing is often not carried out to the required extent.

Even when developers see the urgent need for security products, they often lack the skills to operate various security tools and interpret the results. Learning these skills, in turn, prevents developers from creating new features for their software.

1.4 MANAGERS AND DEVELOPERS MUST BE ABLE TO BE ABLE TO RELY ON EACH OTHER

Managers often don't have the time or expertise to review and assess the security of the software they develop, so they must rely on their software developers to create a secure application. However, developers are under additional pressure to develop one feature at a time.

As a result, they have to rely on the frameworks they use to secure their code and have no security themselves. This only leads to the illusion of a secure application, which is not checked by a continuous process.

2. EXAMPLES OF INTEGRATED SAFETY

It is crucial that security is embedded in existing systems and structures so that the security layer does not hinder users. A practical example is a revolving door that only allows one person to pass at a time. This gives security personnel a clear picture of this part of the building's security, while the doors are not perceived as security measures by their users.

Ease of integration of steps is one of the prerequisites for security measures to be successfully implemented in modern development practices. When WhatsApp introduced end-to-end chat encryption overnight without users noticing, it increased the security of millions of people's communications. As a widely accepted form of communication, there were no delays or additional installations to enable the security feature, so users didn't have to change anything in their application.

Implementing such usable security in software development allows meeting deadlines (software releases are not delayed by security issues). It also improves inter-departmental relations as conflicts between the software development department, and the security department can be more constructive and empower software developers.

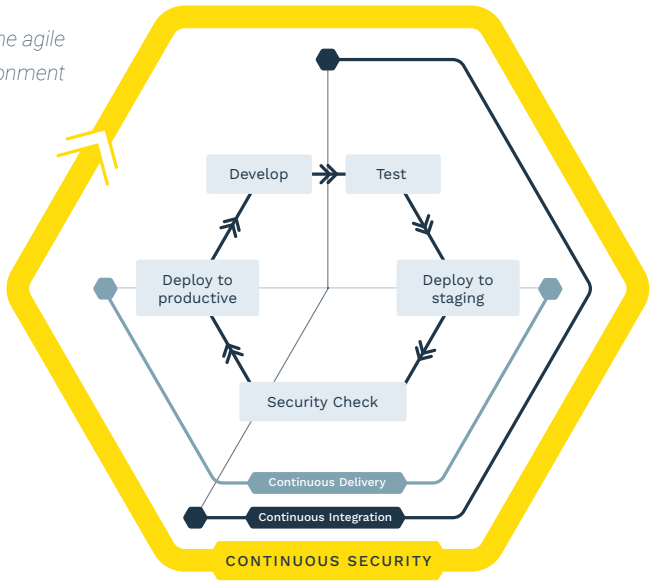


Figure 1: Security in the agile development environment

3. IMPLEMENTATION OF SECURITY IN THE CONTINUOUS INTEGRATION PROCESS

One way to solve the mentioned problems is integrating security tests into the CI/ CD process (see Figure 1). To successfully incorporate such scans into the daily routine, several aspects need to be considered:

+ Safety tests complement other forms of testing
+ Integration into the CI/CD process is essential for continuous security

Security testing must integrate seamlessly with the current development environment so that software developers do not have to leave their familiar environment.

A normal CI/CD process includes a build server that triggers certain functional tests when specific actions occur (e.g., the software is deployed). This may mean that certain unit tests are run each time a push is made to a repository. When a pull request is created or merged, additional tests such as integration tests may be run. The results are usually collected, presented in the build server's user interface, and pushed to different communication media such as a Slack channel.

Implementing functional security tests means that the build server automatically triggers the security tests based on the developers' configuration. Automated tests can additionally be triggered at specific time intervals to ensure continuous security.

One way to implement this is to scan all nightly builds with the automated security scanner. Just as a unit and integration testing have become standard tools for modern software developers, security testing must be fully integrated into their daily workflow.

The security scanner must bundle security scans for the developer and make sense of identified vulnerabilities. This ensures that developers whose core competency is not security engineering are not hampered by the need to integrate multiple tools or interpret command line output from security scans.

4. AUTOMATED SECURITY SCANS STRENGTHEN DEVELOPERS

Instead of being an obstacle for developers, they are supported by adequately integrated security scans. They provide immediate feedback. For example, after a pull request is created and the code is deployed to a test or staging environment. As a result, they still know what code they worked on in the hours or days before and don't have to re-dive into previous work.

Proper testing provides feedback on existing vulnerabilities and provides links to resources or guidance on how to resolve an issue. This level of quality control gives developers back sovereignty over the security of their code. Instead of creating manual security tests, they can focus on their revenue-generating work, like developing new features.

In addition, these tests can free up cognitive resources that are tied up by supervisor pressure. Because a software developer's job is often seen as just a feature producer, (non-tech) managers often see it as a waste of time to put too much work into things like code quality or security.

However, in a security emergency, the immediate responsibility for fixing it lies with the developer who implemented the code that contains the vulnerabilities. An automated security scan gives the developer a great tool to continuously check the security status of the software while focusing on their core tasks.

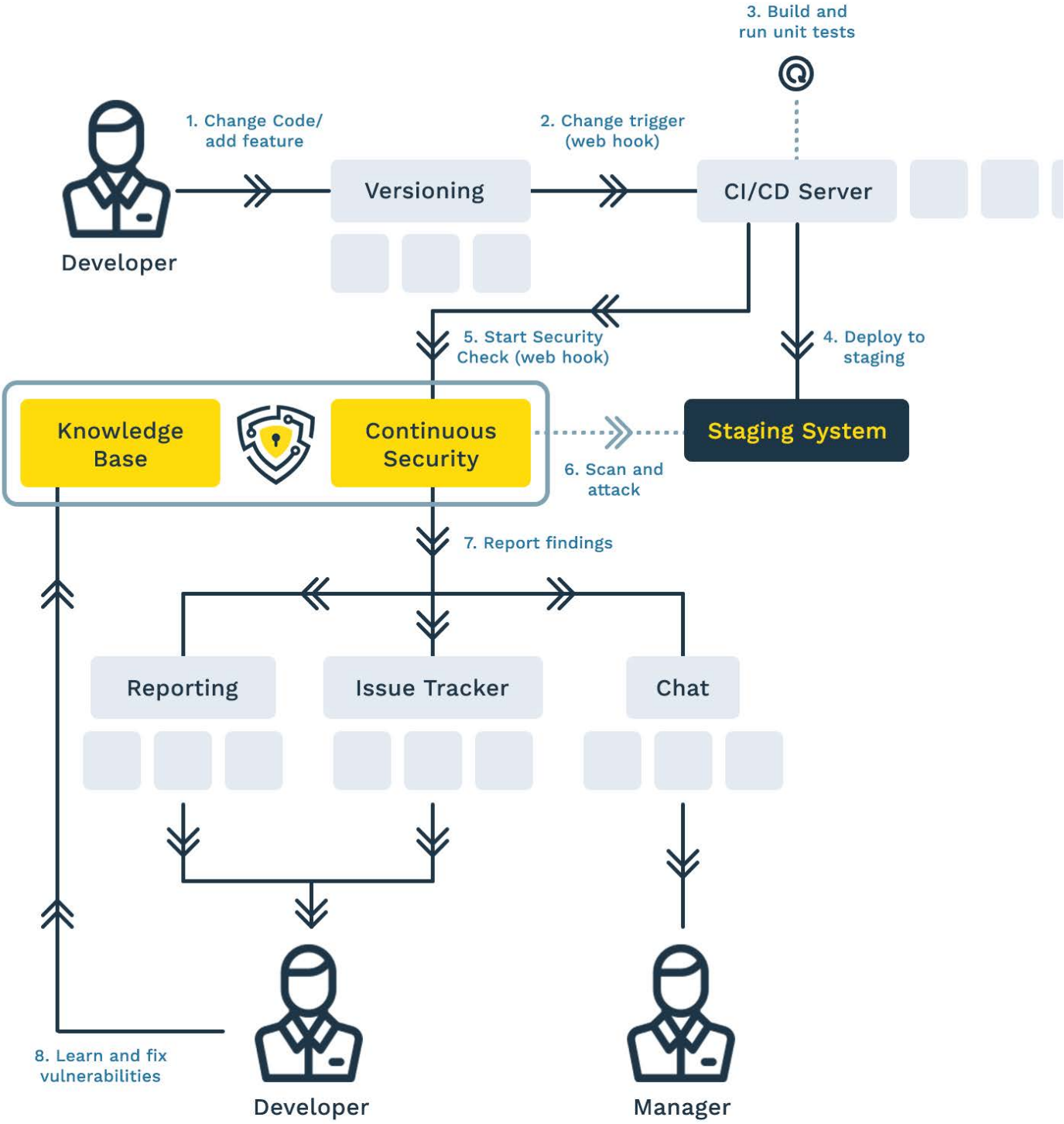
Once automated security is integrated into the daily workflow, developers can independently verify that their code has passed a security scan with each released product version. In addition, a solution that always includes the latest security checks - such as a SaaS security scanner - ensures that any newly discovered vulnerabilities (known as zero-day attacks) are also included in the scan.

+ Continuous security is consistent with applicable security
+ Developers can focus on their primary tasks without ignoring security issues

5. THE CRASHTEST SECURITY SUITE OFFERS CONTINUOUS SECURITY

Crashtest Security detects web application security vulnerabilities in real-time. For this purpose, we have developed a cloud-based security scanner that is offered as a SaaS model. As a result, we ensure that developers can fully concentrate on their revenue-generating work while we detect the vulnerabilities. As software developers ourselves, we know how difficult it can be to produce software with a high user experience and all the features you need - and to do it all securely.

That's why we're writing a toolchain of security scanners that a webhook can trigger. Reporting is done so that developers can focus on solving security problems, and executives can focus on monitoring the software security state. Integrations for existing communication channels, such as Slack, inform managers as needed. If a vulnerability is found, the developer is notified immediately (see Figure 2 for a detailed explanation of the process).


3. Build and run unit tests
1. Change Code/ add feature
2. Change trigger (web hook)
Developer
Versioning
CI/CD Server
5. Start Security Check (web hook)
4. Deploy to staging
Knowledge Base
Continuous Security
Staging System
6. Scan and attack
7. Report findings
Reporting
Issue Tracker
Chat
8. Learn and fix vulnerabilities
Developer
Manager

6. THE CONCLUSION

By being integrated into the development process, similar to what developers are used to, continuous security suites provide security for software that allows programmers to improve the software in terms of security as early as possible.

In this way, developers can provide better software to their customers. The time saved by targeting security issues as they arise (rather than after weeks or months of productive use) allows them to focus on developing new features.

Key points to remember

+ Agile software development requires continuous security.
+ Continuous security must be applied.
+ Using continuous security frameworks frees up resources for developers to focus on their key tasks.

ABOUT CRASHTEST SECURITY

Crashtest Security is a Munich-based IT security company.

As an innovator of cyber security solutions for web applications, it develops automated solutions for vulnerability analysis.

Based on artificial intelligence, vulnerabilities are detected, protection against hacker attacks is increased and transparency for companies, users and developers is created.

Visit our website for more:

WWW.CRASHTEST-SECURITY.COM

CRASHTEST SECURITY GMBH

Leopoldstraße 21
80802 Munich
+49 (0)89 215 41 665

info@crashtest-security.com