Agile Development for Application Security Managers

www.quotium.com

When examining the agile development methodology many organizations are uncertain whether it is possible to introduce application security into the process. In this paper we examine how an organization can develop software by following the agile methodology and still achieve a high level of application security, with minimal adjustments to the process

Agile Development for Application Security Managers

www.quotium.com

Introduction

In today's competitive business environment, it is more important than ever to develop applications not only accurately but quickly. The traditional "waterfall" method is effective, but requires so many steps that the process cannot keep up with today's needs. Agile is a development methodology that speeds up development dramatically, along with several other benefits that make it a popular methodology.

Vulnerabilities in applications pose an ongoing threat to business critical data more than ever before. Organizations are faced with ongoing, persistent threats that originate in their web applications.

Many think that Agile and Application Security cannot co-exist, in other words, that application security is a requirement that Agile development teams cannot meet. Agile development is just too nimble and lean, it cannot be bothered with security, and any attempt to introduce application security into the process will have a great negative impact on the development process.

Having said that, there are organizations that report success with implementation of application security within the agile development process. How can this be achieved? This paper analyzes agile development from the standpoint of application security and suggests what can be done to implement security into the agile development methodology.

Agile Principles Overview

Pure Agile development is defined in the Agile Manifesto. While most organizations do not adopt the pure agile form, many implement a methodology that relies on agile principles. These principles are summarized below.

Responsibility at the hands of the Developers

Development teams are given responsibility and are trusted to get the mission accomplished. The best method of communication is frontal communication among all stakeholders.

Code is Constantly Uploaded and Updated

This allows changes to occur on the fly, allowing the end product to adapt as needed. The life cycle of an agile project consists of nearly constant development and testing, rather than distinct states.

This is the XP (extreme programming) software engineering practice of merging all working copies with the mainline source up to several times per day. This prevents long-term integration problems and as a result the current status of the project is always changing.

Requirements arrive Late in the Process

Agile teams know that requirements will change and evolve through the process, meaning that early investment in documenting requirements is wasted. Early on, there is just enough envisioning of requirements to identify the scope of the project. Firm requirements will be determined deep into the process when the entire team understands better what the end results may be.

Customer-Oriented Requirements

Projects are completed with customer collaboration, the customer ultimately determines the requirements and scope of the project. The ultimate goal is to deliver working software to the customer in a timely manner.

User Stories

A user story is a short, simply-written, statement that explains what a user needs to do as part of the job function of the project. This is one of the facilitators of requirements management.

Automation, Ongoing Testing

Automated testing and test cases allow more frequent testing throughout every step of the agile development process. This allows for quality software even within the shorter development cycles that the agile method produces.

Lean Development

The Agile methodology utilizes the principles of lean manufacturing in the software development process. This minimizes waste, empowers the development team and delivers the finished project as quickly, and affordably as possible.

Business people work together with developers

Daily, ongoing cooperation between developers and stakeholders is the only way to achieve the goals of agile development.

Robust, working software at a constant pace

The key measurement of delivery is working software that does what it should. This should be delivered at a constant pace. The focus is on delivering excellent software.

A process of self-improvement

Periodically, the teams should stop and reflect amongst themselves on problems in the current process and how to resolve these problems.

Achieving Application Security in Agile

In a nutshell, agile development is used to provide quick development and less need for discussion and planning the requirements. This method is used to satisfy customer needs within a shorter period of time. Customer feedback and testing put the product through a number of iteration cycles or sprints until eventually the application meets the goals decided by customer requirements and feedback.

So, you may wonder how secure development and application security testing fits into this process. Waiting until the end of the process can expose or even cause problems that essentially put the project back to square one. So it is important to work security into every step of the process. The following principles show how application security can be achieved, even during fast-paced development that utilizes the agile methodology.

Because development through Agile is a team effort, application security must be a team effort too. Agile is known for its leanness and speed, and in order to implement application security in the process – the same principles ought to be used.

Define Clear Requirements

Development and quality assurance teams are often baffled by requirements presented to them by security. In many cases, developers feel that security procedures are redundant and exist simply to generate more work for the development team, especially when application security testing does not take place until the very end of the project.

Because of this, it is important to define clear expectations, ideally from the very beginning. The development and testing teams need to understand what is the desired level of security and the meaning of achieving this level, as well as which security tests will be conducted and what results are expected. Giving a better description of what the team should focus on in terms of security, information about the testing process itself and the logic behind them can help the developers work security into the project as a whole. Answering the following questions for the developers is a good first step in defining the requirements:

- What are the specific areas of focus in developing securely and testing for security?
- Do these tests replace periodical penetration tests and security audits or are they utilized alongside these testing methods?
- How often should developers test for security and who is responsible for doing these tests?
- What security standards should the development team strive to meet or exceed? (This could be industry standards like OWASP, PCI-DSS, internal organization requirements or something else)

After answering these questions, the development team can best integrate application security into the agile development process from a place of understanding and cooperation, not from a place of suspicion and ambiguity.

It is important to provide not only requirements but support towards the goal of achieving them, whether by training, secure frameworks to work with, etc.

Work with the Process, Not Against it

Agile development teams have certain processes that they utilize during the project lifecycle. Like nearly everything involved with Agile, these processes are meant to be lightweight and lean. The best way to successfully integrate application security into agile development is to work alongside existing interactions. This way the created code meets all the requirements of the project but can also be secured and of high-quality. The Agile

Manifesto states: "individuals and interactions over processes and tools." When it comes to security, you cannot completely forget tools, but you must make certain that the tools you use facilitate interactions between team members by providing common language understandable by everybody, and that they do not encumber the process.

This means for example if a team is using specific bug tracking software – security issues should also be delivered as bugs using the same interfaces. If there is existing automation and regression testing – choose the application security tools that can be integrated into this process. Prefer tools that deliver results in a language developers understand – with location of the vulnerability in the code and preferably with clear explanations and demonstration of vulnerability risk level to avoid arguments over priority of fix.

Essentially instead of making security a separate process that can lead to end-of-project setbacks, security is integrated into every step of the process.

Accommodate Frequent Code Changes

Agile not only involves frequent code changes, it encourages them. As such, it is very important that security testing caters to these same standards. Application security within the agile methodology must also change just as quickly. Application security that provides this need of ongoing security will provide protection without complicating matters for the designers, leading to huge hurdles or forcing these developers to work outside of agile principles.

This means tools that take a long time to run are not effective in these environments, nor are tools that require manual interpretation of results. This is due to the fact it is very likely that by the time the tool is done testing and a reviewer from the security team is done reviewing, the new code would already be in production for days or even weeks.

Create Security Stories

Nearly all teams work with at least some level of delivering requirements as user stories. Presenting application security in the form of user stories keeps the flow familiar and works within the standards of an agile project. Additionally, one of the most important things to remember is that security is part of everyone's job, not something that is pawned off onto one person or team. By formulating security as a user story, it reminds the developers of this important part of their task, and presents security as just another aspect which is part of the development and testing cycle.

Help Create an Agile Application Security Workflow

Explain to Agile developers what is expected in terms of security. Then work directly with them to create a workflow that fits in with current habits, iterations and deadlines. Questions often asked by development teams are:

- who should run security testing, should each developer run on their own code, or maybe have one QA member who is responsible for security testing.
- How often should security tests be performed – should they be on every piece of code or after integration
- Who should the results be delivered to? Development or security?
- Who is responsible for sign off

The meaning of an agile application security workflow is creating a development process with embedded application security throughout all it's phases, while still keeping the agile principles of being lean and quick.

Provide a Training Program

Many developers do not have the training necessary to properly understand application security and perform the testing. Even if a training program is in place, developer turnover makes it very difficult to have everybody always up to date. Before you hand off responsibility to a development team, you should provide enough information to make the process easier.

Use application security tools that involve training in the process. This training will pay off not only for the project at hand, but with future projects developed by a similar method. This is one of the benefits of agile development, future projects are even easier.

Remember that training does not have to mean two weeks of training covering a multitude of topics, some relevant some not, followed by an examination and usually resulting in the developer instantly forgetting most of what they learned. Training can mean for example that if a developer has a specific vulnerability in his code he will need to do a short training on this vulnerability, or training on specific vulnerabilities related to the application they are working on.

Don't be Afraid to make Mistakes and Improve as you Go

Agile development is all about learning as the project proceeds. Any iteration includes improvements and changes, eventually moving towards the end result. The application security process is completed the same way, with changes and improvements along the way.

Conclusion

Secure software, developed by any means, comes from properly testing and following proper security measures. The common perception that agile development methods cannot embrace secure coding practices and application security testing is, for the most part, false. With some flexibility, it is possible to integrate application security within your agile development system.
Developers should pay attention to the risks of not incorporating security within agile development. When software is not properly tested for security, there is a risk of developing insecure software which can lead to data loss and programs that are susceptible to hackers. While there is a cost to security testing, the cost that can occur due to improper testing will generally dwarf this cost.

Application Security as a natural part of the SDLC - Seeker

Quotium's Seeker has been proven to work seamlessly with agile development. Most development teams are usually not experts on security issues. Seeker was developed to make application security available, top notch, and easy. Seeker offers the following:

Application security testing that integrates seamlessly into the development process, integration with functional testing, build servers, and any other existing automation via a powerful out-of-the-box interface. You just put Seeker where you put all the rest of your automatic testing.

- Vulnerable code is highlighted and remediation provided, resulting in minimal work for developers and testers.
- Risks are explained in simple terms, allowing stakeholders to easily prioritize and build a vulnerability management plan
- Both the cause and the fix are shown for developers, allowing to recreate the scenario, see the code that is causing the problem, and get an easy remediation solution, all in one place
- Vulnerabilities are managed as any other bugs
- Speedy testing processes allow full testing of the application in a short time, for new code or as part of regression testing

Using Seeker, an agile development team can implement security during the development process. This offers a huge return on investment as well as improved customer satisfaction, since projects are completed on time and within the constraints of the agile framework.
Seeker is simple and delivers nearly immediate results with little effort or change in the development process.

As stated by Quotium:

"Accurate, clear and simple, Seeker, the best quality application security testing solution maximizes your return on investment to secure your applications."

About Quotium

Quotium brings innovation and quality in software solutions that secure, monitor and improve performance of web applications. In 2011 Quotium pioneered run time code & data analysis technology to detect and mitigate technical and logical vulnerabilities in code. The application security testing software Seeker revolutionized the way web applications are being secured.

About Seeker

Seeker is the run-time code & data analysis application security testing solution for the software development life-cycle. By analyzing application behavior in response to simulated attacks, Seeker detects code vulnerabilities that pose a real threat. It assists in vulnerability management by generating exploits that demonstrate the risk to business critical data. Seeker is the perfect application security testing solution for the SDLC; it can be fully automated and works great in Agile and continuous integration environments.