Research Challenges in Empowering Agile Teams with Security Knowledge Based on Public and Private Information Sources

Michael Felderer and Irdin Pekaric

Department of Computer Science
University of Innsbruck
Technikerstr. 21a, A-6020 Innsbruck, Austria
{michael.felderer,irdin.pekaric}@uibk.ac.at

Abstract. The application of agile methods has become increasingly popular that now it is also used in critical system development. Because of this, it is essential to consider the security aspects during agile development. Despite security being knowledge-intensive, developers and product owners in agile projects still have low security knowledge. To overcome this problem, people require more access to easily processable and up-to-date security information, which should be provided on-time and without excessive effort. In this paper, we propose a framework for security data extraction, processing and application. The framework consists of two main components: a security data collection and analysis component as well as a security knowledge generation component. However, the development and process integration of such a framework poses many challenges that are discussed in this paper.

Keywords: agile development, data extraction, data processing, data analysis, security knowledge security engineering, security

1 Introduction

Agile software development has become increasingly popular to the point that now it is even used for critical system development [1,2,3]. For this reason, agile methods are used in situations where security issues in a system may impact safety. In agile software development, there is a focus on the feature implementation and delivery of value to the customer. Consequently, security aspects are often neglected [4], which may have severe implications for the developed software products or services. To counter this, developers require security knowledge,

In: M.G. Jaatun, D.S. Cruzes (eds.): Proceedings of the International Workshop on Secure Software Engineering in DevOps and Agile Development (SecSE 2017), published at http://ceur-ws.org

which can, for example, be communicated by trainings or guidelines. In a recent survey, Oyetoyan et al. [5] found that the developers' confidence in their software security knowledge is generally low. This is why more effort should be spent on increasing the level of security knowledge in companies. This is especially true in agile software development because there is a strong dependency on people, rather than the processes and documents.

In traditional projects this problem is often addressed by an explicit security officer who is responsible for security issues and transferring security knowledge to developers. However, in an agile development process, where iterations are short and changes are performed continuously, this is often not the case.

When it comes to an agile context, people require *easily processable and up-to-date security information*, meaning it should be provided on-time and without excessive effort. To support this, data can be extracted and processed from both public and private information sources, such as vulnerability databases, forums, blogs, conferences on security, or emails. Afterwards, the security measures should be based on the extracted data. Activities in this direction are currently performed in the cyber threat intelligence sharing [6], where cyber threat information is shared within a community to support IT-security management. The focus of this paper is not on IT-security management, but on supporting agile software development teams. On the other hand, the focus of cyber threat intelligence sharing is on supporting IT-security management, but not agile software development teams. Nevertheless, tactics, techniques and procedures developed for cyber threat intelligence sharing can be reused.

Intelligence is also one of the four domains of the Building Security In Maturity Model (BSIMM) [7]. Intelligence is summarized as "Practices that result in collections of corporate knowledge used in carrying out software security activities throughout the organization. Collections include both proactive security guidance and organizational threat modeling."

In order to empower agile teams with security knowledge, we want to present a framework for security data extraction, processing and application, and related research challenges.

Consequently, the remainder of this challenge paper is structured as follows: Section 2 sketches a framework for security data extraction, processing and application. Section 3 discusses research challenges on the basis of the defined framework. Finally, Section 4 concludes the paper.

2 Framework for security data extraction, processing and application

In this section, we give an overview of a framework for security data extraction, processing and application. Figure 1 provides an overview of the proposed framework.

The proposed framework consists of two major components, i.e., a *Security Data Collection and Analysis Component* and a *Security Knowledge Generation Component*. In the following, we explain these two components in more detail.

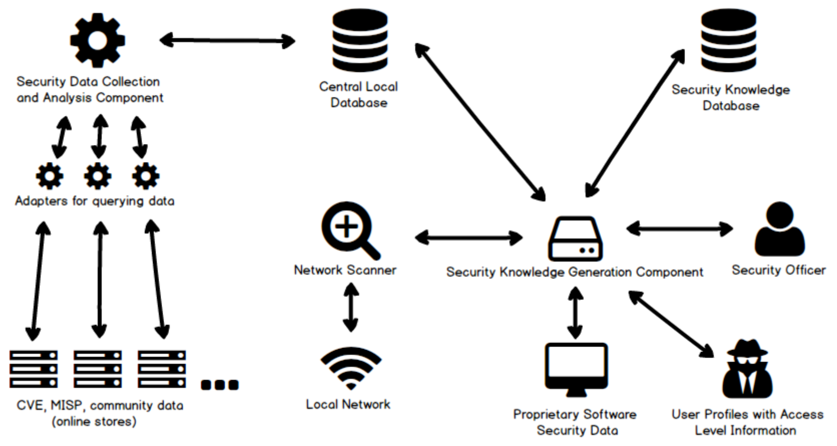
Security Data Collection and Analysis Component
Central Local Database
Security Knowledge Database
Adapters for querying data
Network Scanner
Security Knowledge Generation Component
Security Officer
Local Network
CVE, MISP, community data (online stores)
Proprietary Software Security Data
User Profiles with Access Level Information

Fig. 1. Proposed framework for security data extraction, processing and application

The Security Data Collection and Analysis Component is responsible for the following: data extraction from various data sources, quality assessment of data and data merging in order to provide the data in a processable form.

Currently, there are several available online sources, which store security related information. These include the Common Vulnerabilities and Exposures (CVE) [8] as well as the Malware Information Sharing Platform (MISP) [9]. Many well-known websites, such as Twitter or YouTube, provide APIs that can be used to search for security data. In addition, data can be collected using public sources like security forums and websites. While each of the sources retains the data in different formats, there is no unified approach available for obtaining such valuable information. Thus, it is necessary to create custom extractors for each of the sources. In the framework, data extraction is conducted using multiple adapters, which represent light-weight algorithms whose sole purpose is to retrieve the data and store it into a central local database. They all implement the same interface. Each adapter is categorized into a version, which corresponds to a version of a source. By doing so, it becomes easier to identify which adapter should be plugged to a certain source because the security online stores are regularly updated. Consequently, this allows adapters to be reused and with only minor changes applied to any new versions of security databases.

Once the data is extracted and available, it must be formatted, the quality assessed and then merged. This is done through the Security Data Collection and Analysis Component. Because of the type of information being handled and the fact that there are different data fields to deal with, this is a highly complex task. For instance, the CVE database contains information about vulnerabilities and exposures, while the MISP database consists of malware and threat-related data, and it is expected to have multiple mismatches. In order to overcome such differences, a general format is proposed, which includes information such as name, type, year, target platform, description and reference.

The Security Knowledge Generation Component processes the extracted security information in order to provide it for different roles and various purposes in the agile development process. For instance, a developer can be provided with a security dashboard or concrete guidelines on how code can be secured or security properties can be tested. As for the product owner, they can receive guidelines on security requirements. Finally, when developing a safety critical system, a developer who is responsible for the system architecture can be provided with generated attack models that can be integrated with available system models to perform an integrated safety-security analysis.

An additional input for the Security Knowledge Generation Component is the event data, which is gathered by scanning local networks and proprietary software applications. Because of the fact that there are available software solutions that provide this service, it is unnecessary to reinvent the wheel. It is sufficient to apply Nessus or other similar applications.

Another significant aspect is keeping track of private data sources such as user profiles or email logs. It is important to keep track of users that have access to a project. This includes their access level information and action logs. Based on the available user data, it is possible to apply the user anomaly detection (UAD) algorithms and search for a deviation among different users. If actively monitored and regularly updated, it will help detect possible intrusions via masquerade attacks, which can significantly impact security. In addition, the monitoring of certain private user sources can lead to the enrichment of collected security related information. For instance, custom email scanners could search for a specific set of keywords that are known to potentially impact security. Once they detect a possible security issue, the Security Knowledge Generation Component will generate alerts and reports for the security officer. By doing so, it is possible to provide an active guidance to the agile team members and enforce the required security mechanisms.

All the security related information that were collected using the Security Data Collection and Analysis Component as well as the independent modules are used by the Security Knowledge Generation Component, wherein the data is correlated in order to produce a valuable input for a developer or product owner. The component stores all the generated security information into the knowledge database. This data store is also used to generate additional information by correlating all the resources that are available within. Furthermore, when necessary, a team member can provide additional input for the Security Knowledge Generation Component. The output of the framework includes reports, alerts and charts. Based on the result that is provided by this component, the alerts of different intensities will be generated. For example, if there was an indication that there is a vulnerability that could cause minor damage to a system, such as gaining access to the part of a system where valuable information cannot be retrieved, an alert of low intensity would be generated. On the other hand, if the Security Knowledge Generation Component concludes that the attacker could gain access to valuable data, such as the account information, an alert of high intensity would be created. The framework provides support for 10 types of alert intensities, thereby giving the agile software development team the ability to prioritize security issues and resolve them effectively.

3 Research Challenges

This section outlines relevant challenges in empowering agile teams with security knowledge based on public and private information sources.

In order to develop a beneficial security knowledge database, it is necessary to extract high-quality security information. There are various sources that are available as online data stores. However, the *identification of such sources* is a highly challenging task. In fact, *combining too many sources* may result in an overpopulated database, making it extremely difficult to devise security knowledge due to the diversity of data. On the other hand, the *consolidation of a small number of sources* could result in an inability to devise information that are more valuable than the ones already present in the online security data stores. Thus, to avoid these related issues that may arise, the most appropriate approach is to combine small number of sources and then gradually increase the number when required. Overall, only a small amount (about 5%) of the information found in general sources is somewhat relevant.

Due to the diversity of available sources, it becomes increasingly difficult to *prioritize sources and to assess their validity.* For instance, a certain security database may store data that is considered to be more valuable than another database. Therefore, it is necessary to develop a metric that will determine the suitability of each source.

Another challenge that needs to be addressed is the *extraction of data.* Considering that most of the sources keep data in an unstructured and semi-structured format, it will be necessary to create custom adapters for each specific source.

Once the data is collected, it must be merged, thereby bringing a whole new group of challenges, such as *devising a general format, managing inconsistencies, assessing information that are relative to each other and selecting information based on stakeholder requirements.*

In terms of the *general format for security related information*, it is possible to apply the Structured Threat Information eXpression (STIX) [10], Trusted Automated eXchange of Indicator Information (TAXII) [11] and Cyber Observable eXpression (CybOX) [12], which are formats used to describe cyber-threat information. However, depending on the type of data that will be processed, it might be impossible to transform it to match any of the aforementioned standards. Furthermore, it is to be expected that some data source will have the same or similar data. For this reason, it is essential to *accurately compare and remove any redundancies.* The last challenge to consider when merging security related data is the *field in which the stakeholder conducts the business.* In order to avoid the generation of security artifacts, which will not be of particular value to a stakeholder, it is necessary to apply filters. The filters will iterate through the security data and then remove any unrelated information.

It might be very difficult to fully *automate the proposed framework*. While it is possible to automate the extraction and merging of security related data, the automation of the Security Knowledge Generation Component will be a challenging task. Therefore, the solution is expected to be semi-automated and to provide recommendations for specific roles such as developers. This will add the support for specific tasks such as the definition of guidelines for secure coding. The key prerequisites to ensure continuous security [13] and the continuous consideration of security aspects in all the phases of development are an automated analysis and the recommendation of information for specific tasks.

In order to fully utilize generated security artifacts, it is possible to use them for attack model generation, which would provide an additional input for a security officer. Generated attack models will surely improve the process of penetration testing and might even reduce the tendency of hiring external penetration testers, which is very common in the agile software development life-cycle. However, the *development and generation of attack models* is a formidable task. It requires advanced knowledge in the area of modeling and software security. In addition, it is exceedingly difficult to generate *zero-day attacks* because the security data that describes such attacks is not available. Nonetheless, by correlating multiple generated security artifacts, it might be possible to address this issue.

Finally, the last challenge is the *integration of the proposed framework with the agile software development process*. Since the focus of agile development is on early and continuous delivery, it might be challenging to integrate security related artifacts with the current artifacts. In addition, another obstacle is the lack of general security knowledge of agile teams. Therefore, it is necessary to provide advantageous information at the proper stage of agile software development life cycle.

4 Conclusion

This paper presented a framework for security data extraction, processing and application as well as related research challenges. The framework consists of a security data collection and analysis component as well as a security knowledge generation component. In the future, we will be investigating the stated challenges in the context of agile system development projects.

References

1. Fitzgerald, B., Stol, K.J., O'Sullivan, R., O'Brien, D.: Scaling agile methods to regulated environments: An industry case study. In: Software Engineering (ICSE), 2013 35th International Conference on, IEEE (2013) 863–872
2. McHugh, M., McCaffery, F., Coady, G.: An agile implementation within a medical device software organisation. In: International Conference on Software Process Improvement and Capability Determination, Springer (2014) 190–201
3. Baca, D., Boldt, M., Carlsson, B., Jacobsson, A.: A novel security-enhanced agile software development process applied in an industrial setting. In: Availability,

Reliability and Security (ARES), 2015 10th International Conference on, IEEE (2015) 11–19

4. Cruzes, D.S., Felderer, M., Oyetoyan, T.D., Gander, M., Pekaric, I.: How is security testing done in agile teams? a cross-case analysis of four software teams. In: International Conference on Agile Software Development, Springer (2017) 201–216
5. Oyetoyan, T.D., Cruzes, D.S., Jaatun, M.G.: An empirical study on the relationship between software security skills, usage and training needs in agile settings. In: Availability, Reliability and Security (ARES), 2016 11th International Conference on, IEEE (2016) 548–555
6. Johnson, C., Badger, L., Waltermire, D., Snyder, J., Skorupka, C.: Guide to cyber threat information sharing. NIST Special Publication **800** (2016) 150
7. McGraw, G., Migues, S., West, J.: Building security in maturity model bsimm v6. 0 (2015)
8. Mitre: Common vulnerabilities and exposures. https://cve.mitre.org/
9. Andre, D.: Malware information sharing platform. http://www.misp-project.org/
10. OASIS: Structured threat information expression. https://stixproject.github.io/
11. OASIS: Trusted automated exchange of indicator information. https://taxiiproject.github.io/
12. OASIS: Cyber observable expression. https://cyboxproject.github.io/
13. Fitzgerald, B., Stol, K.J.: Continuous software engineering: A roadmap and agenda. Journal of Systems and Software **123** (2017) 176–189