An Empirical Investigation of Agile Information Systems Development for Cybersecurity

Abdulhamid A. Ardo[10000-0002-0829-570X], Julian M. Bass[10000-0002-0570-7086] and Tarek Gaber[10000-0003-4065-4191]

Department of Computer Science & Software Engineering, University of Salford, Manchester, UK

a.a.ardo@edu.salford.ac.uk

Abstract. Cybersecurity has been identified as a major challenge confronting the digital world, neglecting cybersecurity techniques during software design and development increases the risk of malicious attacks. Thus, there is a need to make security an integral part of the agile information system development process. In this exploratory study, we empirically explore the agile security practices adopted by software developers and security professionals. Data was collected by conducting ten semi-structured interviews with agile practitioners from seven companies in the United Kingdom (UK). The study was conducted between August – November 2020. An approach informed by grounded theory was used for data analysis including Open coding, Memoing, Constant comparison and Theoretical saturation. The security practices identified in this study were categorized into roles, ceremonies and artefacts and mapped onto the different phases of the Software Development Lifecycle (SDLC). We discovered practitioners use five artefacts: security backlog documentation, software security baseline standards, security test plan templates, information security and security audit checklists; and that there are more artefacts than roles and ceremonies. Also, while most practitioners rely on automated tools for software security testing, only one practitioner mentioned conducting security tests manually. These practices that we have identified comprise a novel taxonomy which form the main research contribution of this paper.

Keywords: Agile Security Practices, Agile Information Systems Development, Cybersecurity, Software Security Testing, Security Specialist, Grounded Theory, Automated Test Tools.

1 Introduction

Agile information systems development is increasingly becoming influential, and its main goal is the improvement of software product's quality and productivity [1]. Conventionally, the agile development process consists of roles, ceremonies or practices and artefacts. The three typical agile team roles are the scrum master, product owner and development team members. These roles describe the key responsibilities of agile team members. Agile ceremonies are periodic meetings conducted to check project quality and schedule, while artefacts define the elements adopted for sharing project information [2]. The three popular agile methodologies are extreme programming (XP), scrum and lean. These methodologies, however, do not put much emphasis on cybersecurity issues as there are limitations to their use of security activities [3,4]. Many of the cybersecurity challenges are attributed to lack of integrating security activities during the development process. Also, the risk of unintentionally using insecure practices in most situations leads to the production of insecure software systems. To develop a secure software, practitioners need to adopt many of the quality-improving features of agile software development. Some of these features include iterative development, continuous integration, retrospectives, and constant refraction, among others. It is therefore imperative to integrate security best practices into the agile software development lifecycle to prevent cyberattacks [5]. Backman in [6], have tried to answer the question why software security breaches remain a challenge. By conducting interviews and surveys at a software company he found insufficient knowledge, and inadequate testing policies as some of the issues responsible for software vulnerabilities. Therefore, to address the above enumerated problems of using agile for building secure software, some of the existing studies have highlighted the need to involve the entire agile team and integrate security activities throughout the development process [7-9].

The existing literature have explored agile security as highlighted in this section. Villamizar et. al., [10] have discussed the lack of security requirements integration into agile practices. Bartsch in [11] have explored agile security issues through practitioner interviews. His findings highlighted the need for more focus on improving practitioner's cybersecurity awareness and expertise. Also, the study suggested adequate focus on stakeholders' involvement which is inadequate in security-critical agile projects. Queslati et. al., [12] conducted a systematic literature review to identify software security development challenges reported in literature. The study found 20 challenges, 14 of which are valid problems to the agile research community and the remaining 6 were neither caused due to agile principles or security assurance practices. Rindell et. al., [7] have identified the security activities from OWASP, Microsoft SDL and Common Criteria models and mapped them into agile development processes, practices, and artefacts. While the three studies discussed in this section are related to agile methods security [10-12], none of these studies have developed a taxonomy of agile security practices categorized by roles, ceremonies, and artefacts. Detailed literature review on the intersection of agile and security issues are discussed in the related work section.

Thus, this study aims to fill this gap by improving the security of agile information systems development. In this paper, the main research question we seek to answer is: "What security practices are adopted by practitioners during agile information systems development?". This paper will also answer the following sub-questions:

RQ1: How is software security testing performed in agile teams?

RQ2: How the software security specialist role can be integrated into agile process?

To achieve the research aim, we conducted ten practitioner interviews and adopted a data analysis method informed by the grounded theory. This study contributes to the existing literature in three ways: (i) presenting a taxonomy of empirically identified security practices categorized into roles, ceremonies, and artefacts; (ii) Describing the different ways of software security testing in agile teams; (iii) Describing how the security specialist role can be integrated into the agile process.

The rest of this paper is organized as follows: In section 2, a review of related work is discussed, along with a brief on secure software development models. The paper then introduces the research method adopted for the study in section 3, providing information on the selected research sites, data collection and analysis process used. The paper findings organized into sections on software security testing and integrating the security specialist role in agile process are presented in section 4. Section 5 discusses the study findings in relation to the literature and the research questions. Section 6 concludes the study by explaining the three main research findings.

2 Related Work

This section introduces software security and discusses the prior studies on the intersection between security and agile information systems development.

2.1 Overview of Software Security

The ubiquitous nature of software is bringing enormous benefits to the way human transact their daily activities. However, this also comes with the consequence of increasing system flaws and misuse by malicious users. With the daily reported cases of cybersecurity breaches in the news and social media, software security has attracted the attention of industry players and the academia [13]. The increasing software vulnerability can also be attributed to the astronomic increase in computer connectivity as well as complexity of information systems. The concept of "building security in" is one strategy advocated by researchers towards confronting the challenges of cyberattacks [14].

The term "Cybersecurity" is defined as the set of practices, guidelines technologies and tools used for protecting organizational assets [15]. Cybersecurity needs to guarantee three security attributes which includes confidentiality, integrity, and availability. Most computer security vulnerabilities are attributed to the use of poor programming practices by software practitioners [16]. The CWE/SANS lists the top 25 software errors causing majority of cyber-attacks [17]. Similarly, the Open Web Application Security Project (OWASP) also lists the top 10 web application security risks for managing websites [17]. The purpose of annually publishing the lists is to serve as a guide for practitioners writing codes to be acquainted of current software bugs to avoid. All reported vulnerabilities are usually assigned an identification number and stored in a database known as the National Vulnerability Database (NVD) [18]. Looking at the cybersecurity issues discussed in this section, there is a need to study what efforts have been made in literature to develop security models.

2.2 Secure Software Development Models

Prior study had described the existing software security models and techniques [8]. Three of the popular models are the Microsoft Security Development Lifecycle (SDL) model [19], Software Assurance Maturity Model (SAMM) [20] and the Building Security in Maturity Model (BSIMM) [21]. Out of these models, only the BSIMM was created through empirical observation and real-world data from software security practices. Also, the above listed models were criticized in literature for being theoretical in their approach and adoption strategies [22]. Each of the models have their own peculiar limitations. The Microsoft Security Development Lifecycle (SDL) framework requires huge documentation and is heavy on processes and organization [23]. Also, some practitioners consider Microsoft SDL threat modelling (risk analysis processes) as too expensive to implement in smaller projects. For SAMM, just like other maturity models adopted for agile software development, the checklist contained in the model are not appropriate for all security-context [8]. While for BSIMM, there seems to be lack of empirical research evaluating the model from a scholarly viewpoint [22]. While the aim of this paper is not solely to build a comprehensive security model, our study has added to the scholarly body of empirical research on secure agile information systems development.

2.3 Security and Agile for Information Systems Development

Existing studies have highlighted the lack of security activities in agile information systems development [3,24,25]. Baca & Carlsson in [3] looked at the compatibility of security practices with the characteristics of the agile manifesto. Bansal & Jolly in [26] evaluated security practices with the aim of proposing ways of developing secure information systems processes. The limitation of the study in [3] is that the security enhanced agile development process produced has not been empirically evaluated and compared with other existing development processes. For the study done by Bansal & Jolly in [26], factors such as cost, time and recurrence which may affect the compatibility between two security activities that can help project managers in decision making has not been considered.

Terpstra et. al., [4] conducted a systematic review on security in agile methods which revealed several methods for integrating security in agile methods. The paper went further to corroborate the aim of this study that little is known about security from practitioner perspectives. The study identified three contextual factors important for shaping security in agile projects. These factors are solutions addressing the artefacts, solutions addressing the human factors in agile and solutions addressing the agile process itself [4]. However, the identified solution factors were only focused on people, but nothing was mentioned with regards to tools or other sophisticated methods used. Also, collected data was only based on practitioners' posts on LinkedIn which will not allow for in-depth understanding of the phenomena.

Rindell et. al., [8] conducted a survey of security engineering activities as practiced by some Finnish agile practitioners. The study surveyed their use of 40 security engineering practices and 16 agile development activities. The study observed the discrepancy between the level of use and perceived impact of security activities. However, the study findings were not compared with other existing baseline surveys conducted in other countries.

Baca et.al., [9] proposed an enhanced secure agile development process for a money transfer system at Ericsson Corporation. The study identified security roles which were categorized into four competences including security manager, security architect, security master and penetration tester. However, an obvious drawback of the proposed system is the extra resources required with through the introduction of four different security competences.

Cruzes et. al., [27] studied how software security testing was performed in agile teams across different organizations. The existing literature on software security testing broadly focuses on two areas. The first testing focus area are those done for security services such as confidentiality, integrity, availability, authentication, authorization, and non-repudiation [28]. The second focus area is in the aspect of testing for software resilience to withstand attacks [24]. In the context of secure agile software development, security testing issues include validating if security requirements have been implemented properly to avoid attacks. Also, there are issues of identifying unintended software system's vulnerabilities.

Bezerra et. al., in [29] have grouped the agile security practices based on practitioners' assessment in a particular cyber security organization. Thus, a common finding of agile information systems development studies in the existing literature is that agile methods do sometimes comply with security requirements, but it is faced with the issues of higher cost and slower development due to inadequate agile security processes [8].

Based on the discussions above, no study has created an empirical taxonomy of agile security practices categorized into roles, ceremonies, and artefacts in a single study. Therefore, this gap in knowledge is what our study aims to fill. While there exists a practitioner study that developed taxonomy of agile methods artefacts based on empirical interviews [2]. Also, there is an article that used practitioner descriptions of agile methods tailoring by focusing on the product owner role [30]. Since there is still paucity of empirical evidence on agile methods security in practice and so we think our study can add value to the existing body of knowledge.

3 Method

To answer the earlier enumerated research questions, we conducted ten empirical interviews with selected UK practitioners working in seven companies as shown in Table 1. The research adopted a data analysis method informed by grounded theory methodology. Grounded Theory is a systematic methodology which is aimed at theory construction using qualitative data. Due to the paucity of literature on agile security from practitioner perspective, the Grounded Theory methodology allows for the emergence of concepts grounded in data [31,32]. Also, the Grounded Theory is an appropriate methodology in software engineering for constructing theories relevant to practitioners [24].

3.1 Research Sites

We identified and selected appropriate research sites as shown in Table 1. Due to the Covid-19 pandemic, the first author conducted online interviews with the participants. The snowballing sampling technique was used to select additional research participants. The selected research sites consisted of agile practitioners working in companies operating in different sectors including IT consulting, CRM Company, Healthcare services and the FinTech industry. Most of them are large companies but there are a couple of medium-sized and one small and medium-sized enterprise (SME). The diversity of the research sites provided richness to the data collected and lends credence to the results.

3.2 Data Collection

In this study, the source of data collection was through practitioner interviews. Each of the research participants was sent an information sheet which detailed what the study was about. It further asked for their informed consent to record their responses and indicate their choice of anonymity on a consent form. An interview guide containing topics to be covered was followed. Each participant was asked the same questions even though the wording and sequencing of the questions were not uniform for all participants. The initial interview guide questions were generated from light literature review and the researcher's experience with investigating agile information systems development. The initial questions were subjected to several reviews by the authors. The questions were again modified as data collection progressed following the constant comparison technique of grounded theory. The interview transcribes were manually analyzed at the initial stage before moving to the qualitative data analysis software, Nvivo. The Nvivo software package was chosen because of its ability to facilitate many iterative aspects of the grounded theory process [33]. Nvivo can analyze various types of data i.e., text, audio, video, and photos, whereas tools like Leximancer can only be used for text analysis. Also, the qualitative data analysis tool like ATLAS.ti 4.0 seem to work with data files of limited ranges. This means files must be converted to ASCII format before been inputted.

Table 1: Description of participants' & organizations in the study

Code	Job Title	Software Development Experience (Years)	Business Type	Organization Size
ITCONCco1_SSE1	Senior Software Engineer	10	IT Consulting	Medium-sized
Finco1_SSE2	Senior Software Engineer	16	Financial Services	Large
ITCONCco1_SDA	Senior Developer Analyst	25	IT Consulting	Medium-sized
CRMco1_FSSD1	Full Stake Software Developer	3	Customer Relationship Management	Large
HEALTHco1_SD1	Software Developer	3	Health	Small
ITSERVco1_VP-COS1	Vice President Cyber Operations Security	25	IT Services	Large
LAWENFco1_CSS1	Cyber Security Specialist	4	Law Enforcement	Large
CYBERFOUDco1_ADL1	Cyber Foundry, Analyst Developer Lead	10	Cyber Foundry	Large

3.3 Data Analysis

After conducting each interview, the first author listened to the recordings many times to ensure accurate transcription to avoid distortion in meanings [34]. By adopting the grounded theory methodology for this study, the authors performed the four major activities described by the Glaserian Grounded theory to analyze the collected data. They are (i) reviewing the data to identify repeated themes, (ii) use keywords to categorize the themes, (iii) code the themes and (iv) categorize the themes through the relationships identified. These activities are summarized as open coding, memoing, constant comparison and theoretical saturation.

3.3.1 **Open Coding:** This stage involved line-by-line reviewing of the interview transcripts [32] with the aim of identifying key themes from the interviewee's responses. Going through the first interview, we came up with 42 codes. A second interview was analyzed where 29 codes emerged. Subsequently analyzed transcripts had lesser number of codes as many themes were already identified in the initial interviews.

3.3.2 **Memoing:** In this paper, memos were written to capture interviewee's responses and show the relationship between the different concepts and categories. Brief notes were written on different related topics containing verbatim quotes from interviewees pulled together to make-up a memo. Using the interview transcripts as primary evidence, 6 memos were written out of which 2 are related to the theme of this paper. Memo writing helped elucidate the authors ideas and re-focused further data collection.

3.3.3 **Constant Comparison:** This research used the constant comparison technique to iterate between data collection and analysis. The data collected was constantly compared within itself as well as other instances of same and similar events. The technique was essentially helpful for refining generated codes and categories.

3.3.4 **Theoretical Saturation:** It defines the point in the data analysis process when no new categories emerge, and the data categorization is not impacted by adding more interviews [32].

4 Findings

This section explains the research findings which includes a novel taxonomy of agile security practices developed from the analysis of interview transcribes which have been categorized into roles, ceremonies, and artefacts. Two agile security related memos will be discussed in this section. They memos describe the different software testing methods and tools used in practice. The second memo explains the functions performed by security specialists in the different companies and how they interact with other roles in an agile process.

As earlier mentioned, the key finding of this study is the identification of security practices categorized into roles, ceremonies, and artefacts. We particularly discovered more artefacts than roles and ceremonies. The five artefacts are security backlog documentation, software security baseline standards, security test plan templates, information security and security audit checklists. All the artefacts were mapped in detail onto the stages of the agile SDLC.

The backlog documentation describes all security-related activities contained in the product backlog. According to practitioner (ITSERVco1_VP-COS1) "*All our security activities are prioritized in a sort of security backlog document ... it is normally updated based on a project's peculiar requirements.*" Another important artefact discovered from analyzing empirical interviews is the security baseline standard. "*We adhere strictly to the government-regulated standard requirements like the GDPR ... Additionally, we have our company baseline standards like basic stuff such as encryption and using strong passwords…*" (ITCONCco1_SSE1). These baseline security standards help organizations protect their critical resources such as servers and workstation from cyberattacks.

While baseline standards are sometimes classified as either High-level or Technical, the security test plan template is categorized under the technical group. "*We adopt test plans that will ensure our software operates securely…*" (ITCONCco1_SSE1). The objective of adopting a template for the testing phase is to ensure a process of identifying security threats and the elimination of issues specifically on the safety and integrity of the software. Lastly, the chief technology officers are commonly responsible for issuing the checklist used in their company for information security and audit checks. "*It is part of my duties to issue security audit checklist to the security and development teams…*" (ITSERVco1_VP-COS1).

There are other activities identified which were accordingly mapped to specific ceremonies and roles on the SDLC. The four ceremonies include security sprint planning meeting at the pre-requirements phase, conducting API security meetings at the design stage, conducting secure code review sessions at the implementation phase, and performing security retrospective meeting at the release phase. For the role's activities, we identified security specialist or a times dedicated security team who are involved in all security related discussions from the requirements to the release phases. While we have done a detailed empirical study on practitioner security practices, however, it will be difficult to include the diagrammatic representation of the taxonomy in this paper because of space limitation.

4.1 Software Security Testing

At the testing phase of software development, the use of automated test tools is common among interviewed practitioners. The reliance on these tools indicates a somewhat straightforward absence of other testing methods as reported by this study's general findings. The identified testing tools in this study were categorized into two groups which are the specialized vulnerability testing and standard software testing tools. There are also practitioners that combine manual and automated testing at different stages of software development.

The specialized testing tools mentioned by practitioners in this study were mainly used for vulnerability assessment, scanning and management. According to practitioner (ITCONCco1_SSE1) "*We use a tool known as OpenVAS and what it does is to give* *you a free vulnerability assessment test…*" Basically what the user needs to do is to supply the port numbers and other details and the tool does the testing. Practitioner (ITCONCco1_SSE1) have highlighted the advantage of using the OpenVAS tool where he said, "*OpenVAS is pretty easy and freely available to use as well…*" Apart from the OpenVAS tool, other software companies use software tools such as Qualys, Rapid7 and Nessus to conduct vulnerability assessment. Among the reason's software practitioners mentioned for using these tools are providing specialized functions such as automating network auditing, identifying threat actors through cloud-based solutions, and providing other penetration testing services such as website scanning to identify potential vulnerable spots, confidential data searches and compliance checks as well. According to practitioner (ITSERVco1_VP-COS1) "*vulnerability testing using Qualys or Rapid7 or Nessus takes place at all points, so we know that we are not introducing any vulnerabilities …*" The adoption of automated testing tools in other companies is informed by the diverse capabilities of open-source frameworks such as Kali Linux. According to practitioner (CYBERFOUDco1_ADL1)

> *"We use an array of different tools for security vulnerability checks, most of which can be found in Unix … Kali Linux is one which has a whole platter of tools in that to analyze and exploit software security issues…"*

In standard software testing, GitHub is a common collaborative code hosting platform that allows co-located practitioners to work together on projects. Some practitioners prefer using specialized plug-in on the GitHub platform to perform vulnerability checks. According to practitioner (CRMco1_FSSD1) "*We use a GitHub plug-in called Snyk for vulnerability checks. Previously, we used Greekeeper but recently moved to Snyk because we felt it is a bit better…*" The GitHub platform has various features for things like code verification and modelling of threat actors. This is explained by practitioner (Finco1_SSE2) who said, "*Basically some tools are used for code verification from the security perspective and others for internal tasks to check and identify imposters …*"

There are certain practitioners that favour combining the use of certain testing tools for security checks on parts of their application like APIs with manual software testing. Postman is an API testing tool which enables the automating of various forms of tests such as functional tests, regression tests and end-to-end tests, among others. "*We use Postman to do certain checks on APIs … we use that to set up automated tests to prevent human errors…*" (ITCONCco1_SDA). Moving to the implementation phase, (ITCONCco1_SDA) prefers to perform software test manually. "*I don't use any tools for security tests on codes because coding analysis needs to be done manually by going through the code and working that in…*"

Practitioners involved in security testing who participated in this study discussed very little about the techniques they use for mitigating or managing risks during software testing. According to practitioner (CRMco1_FSSD1), "*We use slow increments and the maker checker approach to develop our projects… You are not allowed to develop, push, approve and merge your code all by yourself...*" Apart from (CRMco1_FSSD1), there are other practitioners that manage risk also by relaying on the iterative feature of agile software methods. "*The short lifecycle and iterative way of* *development minimizes the risk of project delay by been able to provide exactly what the customers requested*" (Finco1_SSE2). The second agile risk management technique described by practitioners in this study is a method where a company uses the developed software internally before pushing it out known as "Dogfooding".

> "*We dogfood a lot of our products internally… We have like a lot of Sales and Support Staff who use our products internally first before it is pushed out and that helps us mitigate risk…*" (CRMco1_FSSD1).

4.2 Security Specialist Role

Typically, agile teams are composed of three roles which are product owner, scrum master and self-organizing team members. In this study, we have discovered another role known as the security specialist. "*In my organization we have an expert that handles issues related to security design and architecture and also responsible for implementing it…*" (CRMco1_FSSD1). The security specialist is responsible for handling all security related tasks, but their function sometimes overlaps with that of the product owner to ensure that the information systems development process does not impose any security risks. Practitioner (CYBERFco1_ADL1) mentioned that their security officer does more than just handling security but performs other tasks for the company. "*So, the security guy is also involved in software development … the person does PR and some other assigned tasks for us…*".

There are situations where a company has a dedicated team that handles all cybersecurity related issues of the information systems development process. According to practitioner (Finco1_SSE2), "*we have a group of experts when it comes to security to act like internal hackers to try to expose any holes in the system…*". Thus, security roles vary across different projects and organizations. During the data collection phase of this study, we found that a lot of decisions about security are mostly handled by those having security tasks assigned to them. According to practitioner (CYBERFco1_ADL1) "*Security is handled by …… practitioners having job titles like security specialist…*". While having a security specialist within the software development team is one way of doing it, in some other organizations all stakeholders are involved in security decision as explained by (ITSERVco1-COS1), "*All stakeholders are required, you got the Business, the Development Lead, Legal team to handle legal requirements that might need to be met…*" (ITSERVco1-COS1). Since security is not something done in isolation, it will be better done with all the key stakeholders. Security decisions are mostly considered as business decisions because there are some financial impacts to them.

At the security requirements gathering phase, a better way of developing secure application will be to involve all stakeholders. Therefore, all the stakeholders need to know the threats and vulnerabilities in their domain. Practitioner (ITSEVco1-VP-COS1) indicated that when he said:

> *"So, it's engaging the right stakeholders at the right time and it's not just a security talking to a techy and saying these are the requirements. It needs to be understood across the board…"*

At the design phase, the data collected did not point to an exclusive security role. However, the software security specialist and software developer roles traverse from the requirements to the release phases of the development lifecycle.

5 Discussion

The novel taxonomy developed in this study has categorized the security practices into roles, ceremonies, and artefacts. The taxonomy has helped to structure the identified secure agile practices. While our study identified eight security practices at the implementation phase, there is a published work that showed only three practices at the same phase [29]. Our taxonomy will help to improve organization's security activities as it encompasses different practices (roles, ceremonies & artefacts) rather than previous research endeavor which focused only on the security roles in a team [9].

5.1 Software Testing Methods used in Practice (RQ1)

To answer the research question "How is software testing done in agile teams?", we found that almost all the practitioners use testing tools which were categorized as either security specialized or standard software tests. Only one practitioner stated his preference for manual software testing rather than using security test tools. Our study findings show that there is a consensus among the participants involved in our study on their use of automated testing tools. Thus, the findings reveal how practitioners have adopted different specialized testing tools to identify and prevent attacks to their software systems. Comparing our findings with the literature, a positive security trend exists due to the extensive use of security tools and reliance on automation [8].

In practice, the agile testing quadrants developed by Crispin and Gregory is popularly used [27]. Our study findings have shown that software companies rely on two of the quadrants which focuses on manual testing and the use of testing tools. For the testing tools quadrant, some of the tests underneath it includes security, performance, or load and illity testing. While the manual testing quadrant has test such as user acceptance testing (UAT), usability and exploratory testing. However, none of the practitioners mentioned anything about adopting illity software testing or the tools used. Existing empirical studies have shown that practitioners heavily rely on specialized testing tools to verify developed software systems [8]. Other testing phase activities mentioned in literature include test cases, but penetration testing is adopted to a lesser degree [8]. However, practitioners involved in our study mentioned adopting specialized tools for security testing. This might be because penetration testing is becoming more common nowadays with the increased number of cyberattacks.

Apart from adopting automated tools, an existing study have emphasized the need to adopt risk management approaches to drive the security testing process [27]. While there is a need for further studies to understand how risk management techniques can be applied to security projects that have adopted agile methods, practitioners in this study only mentioned techniques such "dogfooding" and iterative development approach. The adopted techniques need to be properly understood and their structure well-conceptualized to protect against threats arising from risky activities.

5.2 Integration of Security Specialist Role into Agile Process (RQ2)

The second question posed in this study was to discover "How the software security specialist role can be integrated into agile process?", we discovered from the practitioner interviews conducted that security specialist are involved in different activities depending on their organization and operational environment. We have seen that organizations constrained by budget assign security duties to a single security personnel. Riisom et. al., in [5] have however highlighted the drawback of the security specialist role as sometimes not involved in daily project activities and that other team members might not have the necessary skills to fix software security flaws. There are other software development companies in this study that instead have security teams composed of various roles responsible for managing project's security resources. While a security team ensures better management of security resources on one hand, additional roles increase project costs [26]. Backman in [6] showed the relationship between developer security awareness and their involvement in penetration testing activities. Since developers are assumed to have some security knowledge, adopting secure testing guidelines such as OWASP in addition to training can improve security knowledge levels of practitioners.

6 Conclusion

One of the objectives of agile methods is to improve quality and responsiveness during the development process. However, we have found there tends to be a gap between agile and security practitioners thinking. Our study has provided a taxonomy of agile security practices categorized into roles, ceremonies and artefacts and mapped onto the SDLC, which was derived from empirical practitioner interviews Our exploratory study comprised ten semi-structured, recorded, and transcribed interviews to better understand practitioner perceptions of agile security issues. Our study made three main findings as follows.

Firstly, we were surprised to discover that our taxonomy shows that there are more artefacts than roles and ceremonies. We identified three ceremonies which included security sprint meeting, secure APIs, and security retrospectives. We further identified two security roles which are the security specialist and penetration tester. The five artefacts we identified are: security backlog documentation, software security baseline standards, security test plan templates, information security and security audit checklists.

Secondly, our study discovered that more security practices identified fit onto the implementation and verification phases of the SDLC. Again, we were surprised that the practitioners in our study did not discuss more practices in the requirements and design stages. This finding is different to what was discovered in previous literature where more security practices fitted onto the requirements and implementation phases.

Thirdly, it is important to point out the importance attached to using testing tools by practitioners. Given that software companies in the UK used specialized automated tools to test for security and software performance, there was only one practitioner that expressed his preference for manual software testing. Our study also highlighted iterative development and dogfooding as the techniques used by practitioners for risk management. Finally, this study confirms earlier studies that mention the use of iterative development by practitioners for risk management and mitigation. We also identified an interesting practice where a group of practitioners are sometimes assigned the task of using the software in-house before pushing it out to clients commonly referred to as “dogfooding”.

In the future, we propose to expand the study beyond UK practitioners to examine a developing country context specifically Nigeria. We aim to empirically explore agile security practices and the software security testing techniques adopted by practitioners in Nigeria. We plan to evaluate the security testing techniques identified to support the adoption of secure agile practices for the Nigerian software industry.

References

1. Dingsøyr, T., Nerur, S., Balijepally, V., Moe, N.B.: A decade of agile methodologies: towards explaining agile software development. The Journal of Systems and Software 85, 1213-1221 (2012).
2. Bass, J. M.: Artefacts and Agile Method Tailoring in Large-Scale Offshore Software Development Programmes. Information and software technology, 75, 1-16 (2016)
3. Baca, D., & Carlsson, B.: Agile Development with Security Engineering Activities. In: International Conference on Software and Systems Process, pp. 149–158. Association for Computing Machinery (ACM), New York, NY, United States, (2011).
4. Terpstra, E., Daneva, M., Wang, C.: Agile Practitioners’ Understanding of Security Requirements: Insights from a Grounded Theory Analysis. In: 25th International Requirements Engineering Conference Workshops (REW), pp. 439-442. IEEE, Lisbon, Portugal, (2017).
5. Riisom, K. R., Hubel, M. S., Alradhi, H. M., Nielsen, N. B., Kuusinen, K., Jabangwe, R.: Software security in agile software development: A literature review of challenges and solutions. In Proceedings of the 19th International Conference on Agile Software Development, pp. 1-5. ACM, Porto, Portugal, (2018).
6. Backman, L. Why is security still an issue? A study comparing developers’ software security awareness to existing vulnerabilities in software applications. In: Research Thesis, Linkoping University, Linkoping, Sweden, (2018).
7. Rindell, K., Hyrynsalmi, S., & Leppänen, V.: Fitting Security into Agile Software Development. In: Research Anthology on Recent Trends, Tools, and Implications of Computer Programming, pp. 1026-1045. IGI Global, (2021). https://dx.doi.org/10.4018/978-1-7998-3016-0.ch047
8. Rindell, K., Ruohonen, J., Holvitie, J., Hyrynsalmi, S., Leppänen, V.: Security in Agile Software Development: A Practitioner Survey. Inf. and Soft. Technol, **131**, 106488. Elsevier, (2020).
9. Baca, D., Boldt, M., Carlsson, B., Jacobsson, A.: A novel security-enhanced agile software development process applied in an industrial setting. In: 10th International Conference on Availability, Reliability and Security, pp. 11-19. IEEE, Toulouse, France (2015).

10. Villamizar, H., Kalinowski, M., Viana, M., Fern´andez, D.: A Systematic Mapping Study on Security in Agile Requirements Engineering. In: 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 454–461. IEEE, Prague, Czech Republic (2018). https://doi: 10.1109/SEAA.2018.00080
11. Bartsch, S.: Practitioners' Perspectives on Security in Agile Development. In Sixth International Conference on Availability, Reliability and Security, pp. 479-484. IEEE, Vienna, Austria (2011).
12. Oueslati, H., Rahman, M. M., ben Othmane, L., Ghani, I., Arbain, A. F. B.: Evaluation of the Challenges of Developing Secure Software Using the Agile Approach. International Journal of Secure Software Engineering (IJSSE), 7(1), pp.17-37. IGI Global, (2016).
13. Amoroso, E.: Recent Progress in Software Security. IEEE Software, 35(2), pp.11-13. IEE, (2018). https://doi: 10.1109/MS.2018.1661316
14. McGraw, G.: Software Security: Building Security In (1st ed.). Upper Saddle River, NJ: Addison-Wesley Professional, (2006).
15. Von Solms, R., Van Niekerk, J.: From information security to cyber security. computers & security, 38, pp.97-102. Elsevier, (2013).
16. Stallings, W., Brown, L., Bauer, M. D., Bhattacharjee, A. K.: Computer Security: Principles and Practice: Pearson Education Upper Saddle River, NJ, USA (2012).
17. Siavvas, M., Tsoukalas, D., Jankovic, M., Kehagias, D., Tzovaras, D.: Technical debt as an indicator of software security risk: a machine learning approach for software development enterprises. Enterprise Information Systems Journal, 15(9), pp.1-43, (2020).
18. Zhang, S., Caragea, D., Ou, X.: An empirical study on using the national vulnerability database to predict software vulnerabilities. In: International conference on database and expert systems applications, vol. 6860, pp. 217-231. Springer, Berlin, Heidelberg, (2011). https://doi.org/10.1007/978-3-642-23088-2_15
19. Howard, M., Lipner, S.: The security development lifecycle (Vol. 8). Microsoft Press, Redmond (2006).
20. OWASP SAMM Project.: Software Assurance Maturity Model (SAMM): A guide to building security into software development - v1.5. Technical Report Version 1.5. 72 pages. (2017). https://owaspsamm.org/
21. McGraw, G.: Software security and the building security in maturity model (BSIMM). Journal of Computing Sciences in Colleges, 30(3), 7-8. Evansville, IN, United States, (2015).
22. Rindell, K., Ruohonen, J., Hyrynsalmi, S.: Surveying secure software development practices in finland. In: 13th International Conference on Availability, Reliability and Security, pp. 1-7. ACM, Hamburg, Germany (2018).
23. Rindell, K., Hyrynsalmi, S., Leppänen,: A comparison of security assurance support of agile software development methods. In: Proceedings of the 16th International Conference on Computer Systems and Technologies, pp. 61-68. ACM, Dublin, Ireland (2015).
24. Adolph, S., Hall, W., Kruchten, P.: Using Grounded Theory to Study the Experience of Software Development. Empirical Softw. Eng. **16**(4), 487-513. (2011)
25. Oueslati, H., Rahman, M.M., ben Othmane, L.: Literature Review of the Challenges of Developing Secure Software Using the Agile Approach. In: 10th International Conference on Availability, Reliability and Security, pp. 540–547. IEEE, Toulouse, France (2015).
26. Bansal, S.K., Jolly, A.: An Encyclopedic Approach for Realization of Security Activities with Agile Methodologies. In: 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), pp. 767–772. IEEE, Noida, India (2014).

27. 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, pp. 201-216. Springer, Cham. (2017)
28. Felderer, M., Büchler, M., Johns, M., Brucker, A.D., Breu, R., Pretschner, A.: Chapter One - Security testing: A survey. In Advances in Computers (Vol. 101, pp. 1-51). Elsevier. (2016)
29. Bezerra, C. M. M., Sampaio, S. C., Marinho, M. L.: Secure Agile Software Development: Policies and Practices for Agile Teams. In International Conference on the Quality of Information and Communications Technology, pp. 343-357. Springer, Cham. (2020)
30. Bass, J. M.: How Product Owner Teams Scale Agile Methods to Large Distributed Enterprises. Empirical Software Engineering, 20(6), 1525-1557. (2015)
31. Glaser, B. G.: Theoretical sensitivity: Advances in the methodology of grounded theory. 1978. New York: Sociology Pr. (1967)
32. Corbin, J. M., Strauss, A.: Grounded theory research: Procedures, canons, and evaluative criteria. Qualitative sociology, 13(1), 3-21. (1990). https://doi.org/10.1007/BF00988593
33. Hutchison, A. J., Johnston, L. H., Breckon, J. D.: Using QSR-NVivo to facilitate the development of a grounded theory project: an account of a worked example. In: International journal of social research methodology, 13(4), 283-302. (2010)
34. Oates, B. J.: Researching information systems and computing. Sage. (2005).