2019

AGILE AND SECURE SOFTWARE DEVELOPMENT: AN UNFINISHED STORY

Dave Bishop
Dakota State University

Pam Rowland
Dakota State University

Recommended Citation
Bishop, D., & Rowland, P. (2019). Agile and secure software development: An unfinished story. Issues in Information Systems, 20(1).

Issues in Information Systems

Volume 20, Issue 1, pp. 144-156, 2019

AGILE AND SECURE SOFTWARE DEVELOPMENT: AN UNFINISHED STORY

David Bishop, Dakota State University, david.bishop@dsu.edu
Pam Rowland, Dakota State University, pam.rowland@dsu.edu

ABSTRACT

Given the widespread adoption of agile methods and the rising number of software vulnerabilities, we analyze the literature with an interest in the effect of security practices on software development agility. We propose a novel taxonomy to systematize the body of knowledge around secure agile development and then organize and summarize the selected research using the new taxonomy. At a high-level we create two categories, Phase Focused and Phase Independent. The Phase Focused category is then subdivided along the traditional SDLC phases. The Phase Independent category spans all phases of the SDLC or is phase independent. We conclude that, although there is a significant body of literature on the topic, the story is unfinished. There is further investigation needed to ensure agility as secure development practices are adopted and in regard to empirical evaluations of the proposed agile and secure software development integration approaches.

Keywords: Agile, security, software development, systematic literature review

INTRODUCTION

Agile is the predominant methodology in use in industry today with 52% of companies reporting that more than half of their teams are using agile practices (VersionOne, 2018). At the same time, developing secure software is extremely important given the pervasive spread of security exploits. NIST reported over 16,000 software vulnerabilities across the industry in 2018 (NIST, 2019). Agile and secure software development do not always share the same philosophy and techniques. However, significant work on enhancing agile with secure development has been done resulting in a large number of research articles. Our systematic literature review not only summarizes the literature to date, it also provides a novel taxonomy to systemize the available literature. In addition to the taxonomy, our findings indicate that the story is not finished, and work is needed to ensure that the secure software development practices do not jeopardize software development agility.

Two concepts need to be clarified at the outset. First, this study focuses on agile software development methods. Conboy (2009) provides a well-received description of agile software development methods as, "the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment." The classic and original description of agile values and principles can be found in the Agile Manifesto (Beck, 2001). L. Williams (2012) shows that the Agile Manifesto has continuing relevance in developers' thinking. Comparing agile with traditional software development methods reveals that agile emphasizes adaptive planning, delivered software as a measure of progress and values people over processes, while traditional methods, such as Waterfall, emphasize predictive planning, processes and communication via documents (Balijepally, Mahapatra, & Nerur, 2006). Second, we describe security in the setting of software development. The secure software development concept is to intentionally employ processes and techniques that assess security risks and provide assurance that mitigations to those risks have been applied (Adelyar, 2015; Adelyar & Norta, 2016; Maier, Ma, & Bloem, 2017).

After reviewing over 70 relevant articles, we determine that there still exists a challenge to reconcile the lightweight philosophy of agile methods with the traditional heavyweight nature of secure software development practices, especially using empirical evidence. To define the scope of the research, we chose the terms agile and scrum as representative terms for the development methodology. We chose agile because it generically represents a wide variety

of methodologies of interest. We chose Scrum because it is the most widely used agile method in industry (VersionOne, 2018). In addition, we added the search term security to discover the literature that relates agile/scrum software development to secure software development. By secure software, we mean software that is developed with the intention of reducing security vulnerabilities (OWASP, 2018). We find that although there is a significant body of literature, work still remains on achieving agility while addressing secure software outcomes.

The rest of the paper is organized as follows: first, we will describe our research methodology and data collected, next we will provide an analysis of the results of the selected literature, and finally we will offer a summary.

RESEARCH METHODOLOGY

Our research method is a systematic literature review. We identify and interpret available research related to the current state of secure agile software development which is consistent with the purpose of systematic literature reviews (Kitchenham, 2004). We followed a modified version of the protocol described in Kitchenham (2004). Our research question focused on the state of secure software development when using agile software development methods. We defined two sets of search terms: first, agile and security; second, Scrum and security. We chose the following databases for our search: ABI/INFORM Collection, ACM Digital Library and IEEE Xplore Digital Library. We also did an exploratory search using Google Scholar. Due to the number of articles returned by Google Scholar we anecdotally selected a few articles.

In all cases, we limited our selected articles to those that contain elements on agile software development and include items related to developing secure software. As an additional filter criterion, selected articles must have immediate full text access through our university's library. Searches were performed on Feb-24-2018, Mar-5-2018 and on Dec-26-2018.

After downloading the articles in PDF format, we then imported the corpus into Atlas.ti. We used Atlas.ti as a database to manage the integrity of the data, and to organize our initial analysis. We utilized open coding techniques from Grounded Theory to initially analyze and conceptualize information from the selected articles (Corbin & Strauss, 1990). Grounded Theory utilizes the concept of "incidents" in the raw data, which are tagged (coded) using a conceptual word or phrase representing the meaning of the raw data. We used Atlas.ti to manage our codes and track the association between the raw text and the conceptualized codes. Next, we abstracted the initial codes into a taxonomy of categories and subcategories which are described in the following section.

RESULTS

To organize our findings, we developed a novel taxonomy to structure the growing body of literature around secure agile software development. At a high-level, we see two overarching groups of literature. One set is aligned with individual software development lifecycle (SDLC) phases. We call this the Phase Focused category. The other high-level category summarizes the literature that is not particular to any single SDLC phase or spans many phases of the SDLC. We call this category Phase Independent.

Within the Phase Focused category, we see six different stages that authors have addressed. The requirements phase has received significant attention. The other five categories addressed by the literature are threat modeling, architecture, implementation, testing, and deployment. These five categories have not received near the attention as the requirements phase, as is shown by the counts in Figure 1. We also list the research by category.

	Phase Focused					
	Requirements	**Threat Modeling**	**Architecture**	**Implementation**	**Testing**	**Deployment**
Counts	23	3	2	1	4	2
Articles	Ali (2016), Arbain et al. (2014), Avizienis et al. (2004), Azham et al. (2011), Bowen et al. (2018), Brady (2006), Daneva and Wang (2018), Farid (2012), Ge et al. (2007), Ghani et al. (2014), Knauss et al. (2017), Kongsli (2006), Kongsli (2007), Mead et al. (2008), Othmane and Ali (2016), Page et al. (2007), Peeters (2005), Raschke et al. (2014), Sachdeva and Chung (2017), Siponen et al. (2005), Terpstra et al. (2017), Villamizar et al. (2018), Wang et al. (2018), Yu and Le (2012)	Galvez and Gurses (2018), Shuiabu et al. (2015), Tondel et al. (2008)	Chivers et al. (2005), Tappenden et al. (2005)	Hassan et al. (2018)	Choliz et al. (2015), Kongsli (2007), Othmane et al. (2014), Tappenden et al. (2005)	Clark et al. (2014), Maria et al. (2015), Williams (2018)

Figure 1. Phased Focused Categories

In the Phase Independent category, we identify five specific segments. One segment that has received significant attention is what we classify as "Comprehensive Methodology." This segment develops agile security practices and techniques that address multiple development phases but are not aligned with any particular security framework. An equally active segment is "Security Framework Integration." Numerous authors have provided research in the area of combining techniques and practices from existing software security frameworks with agile software development approaches. There are also some systematic literature reviews that have addressed agile security. Another segment is security training as a means to enhancing secure software development on agile teams. Finally, there are a few articles related to secure agile software development that we categorized in a segment titled "Other," since they covered aspects that did not align with our primary groupings. Examples from the "Other" category are articles that identify the challenges of integrating security practices with agile practices. Figure 2 summarizes the Phase Independent research.

	Phase Independent				
	Comprehensive Methodology	**Security Framework Integration**	**Training**	**Systematic Lit Review**	**Other**
Counts	10	10	3	4	4
Articles	Bansal and Jolly (2014), Barbosa and Sampaio (2015), Bartsch (2011), Beznosov and Kruchten (2004), Ge et al. (2006), Keramati and Mirian-Hosseinabadi (2008), Rajba (2018), Siponen et al. (2005), Singh (2018), Subedi et al. (2016)	Baca and Carlsson (2011), Dorca et al. (2016), Maier et al. (2017), Moyon et al. (2018), Rindell et al. (2015), Rindell et al. (2017), Rindell et al. (2018), Sonia and Banati (2014), Sullivan (2010), Wayrynen et al. (2004)	Oyetoyan et al. (2016), Poller et al. (2017), Rajba (2018)	Oueslati et al. (2015), , Rindell et al. (2017), Shuaibu et al. (2015), Villamizar et al. (2018)	Adelyar (2015), Alnatheer et al. (2010), Alnatheer (2014), van der Heijden et al. (2018)

Figure 2. Phase Independent Categories

In the following sections we summarize by high-level category and sub-category the literature reviewed.

PHASE FOCUSED

Requirements

There are concerns about the lack of focus on non-functional requirements, such as security and safety in agile approaches (Farid, 2012; Knauss, Liebel, Schneider, Horkoff, & Kasauli, 2017; Terpstra, Daneva, & Wang, 2017). Some research suggests that there is a mismatch between security methodologies and agile methodologies (Kongsli, 2006; Wang, Gupta, & Niu, 2018). Security is a non-functional requirement of a software system and includes

confidentiality, integrity, and availability (Avizienis, Laprie, Randell, & Landwehr, 2004). The understanding of how to handle security within agile is still vague (Villamizar, Kalinowski, Viana, & Fernández, 2018). Key challenges for developers are early validation and time-to-market (Raschke et al., 2014). Efforts are being made to integrate security practices within agile, however there is a fear that security will be ignored during development or retrofitted late in the software development process (Sachdeva & Chung, 2017). Teams lack knowledge and training in this area, and every sprint should require a security review (Bowen, Hinchey, Janicke, Ward, & Zedan, 2018).

A variety of requirements-oriented practices have emerged in the literature. One of the primary techniques to date is misuse stories, sometimes referred to as evil stories or abuser stories (Bowen et al., 2018; Daneva & Wang, 2018; Kongsli, 2006; L. B. Othmane & Ali, 2016; Page, Dixon, & Choudhury, 2007; Peeters, 2005; Sachdeva & Chung, 2017). Significant work has been done in regard to Security Backlogs (SB) as an addition to the product backlog (Arbain, Ghani, & Kadir, 2014; Brady, 2006; Ge, Paige, Polack, & Brooke, 2007; Ghani, Azham, & Jeong, 2014; Siponen, Baskerville, & Kuivalainen, 2005). Researchers found that SB improved agility in Scrum and that it was quite feasible (Azham, Ghani, & Ithnin, 2011; Ghani et al., 2014). Two research teams use the Security Quality Requirements Engineering (SQUARE) methodology and SQUARE+R (adds risk) frameworks to address security requirements in agile development environments (Mead, Viswanathan, & Padmanabhan, 2008; Yu & Le, 2012). This area of the SDLC is the most promising for agile software development. The literature utilizes popular agile artifacts such as user stories and product backlogs.

Threat Modeling

In a systematic literature review on web application development security, Shuaibu, Norwawi, Selamat, and Al-Alwani (2015) note that Threat Modeling is the highest frequency security technique which appeared in 66% of their selected articles. According to Shuaibu et al. (2015), Threat Modeling was also in widespread use among agile teams. Galvez and Gurses (2018) identify 21 challenges to using Threat Modeling in an agile and service-oriented software development environment. Although for some challenges, mitigation opportunities are suggested, on a whole they determine that further work must be done to fully integrate Threat Modeling in an agile software development environment (Galvez & Gurses, 2018). Tondel, Jaatun, and Meland (2008) provided a case study on using Threat Modeling in the form of the Elevation of Privilege game (EoP). They found that Threat Modeling via EoP had little effect on the actual software that was developed (Tondel et al., 2008). The integration of Threat Modeling in agile is still in need of further research and empirical support for effectiveness (Galvez & Gurses, 2018; Shuaibu et al., 2015).

Architecture

Architecture has traditionally been associated with top-down approaches in software development. However, there has been some research using iterative security architecture in agile software development projects. One architectural approach involves a "do the simplest thing" approach, developed incrementally and is compatible with XP practices (Chivers, Paige, & Ge, 2005). Architecture incorporates security features into the design and is useful for scaling agile approaches. This lightweight architectural approach provides the grist necessary for security evaluations (Chivers et al., 2005). Another approach proposes an architecture based automated testing methodology to integrate security with agile practices (Tappenden, Beatty, Miller, Geras, & Smith, 2005).

Although there is limited literature on this topic, what is available is positive in that it aligns well with agile processes. Additional work in this area would be helpful to flesh out the agile nature of the practices and to illustrate the security aspects and then substantiate them with empirical evidence.

Implementation

Hassan, Mubashir, Shabir, and Ullay (2018) provide a review of secure software techniques primarily focused on the software construction phase. Although they reference agile software, their discussion does not take into account the short iterations indicative of agile approaches or any other particularly agile concepts. They simply list security tasks such as static code analysis, bug tracking, and exception handling.

Their process does not align well with the lightweight and time constrained nature of agile software development. This is an area that would benefit from additional work identifying secure practices that are in harmony with agile practices.

Testing

Software security testing in agile takes place in a variety of SDLC phases. Tools, misuse cases, and assurance cases are used as inputs to security testing. A case study by Chóliz, Vilas, and Moreira (2015) was used to explore a software engineering team moving from waterfall to agile methods. The security testing was based on Microsoft's SDL for Agile and based on the assessment of four projects. The authors deem the approach a success (Chóliz et al., 2015). Kongsli (2007) applies misuse stories, the OWASP Top Ten, and testing tools such as Selenium's GUI testing tool to automate security vulnerability testing. Another interesting approach in the testing arena is Tappenden et al. (2005). The authors bypass the GUI layer and use HTTPUnit to integrate security into agile processes. Finally, L. Othmane, Angin, Weffers, and Bhargava (2014) propose the use of assurance cases to ensure comprehensive security implementation of security features. Testing appears to be a well-developed integration of agile and security, especially when automated testing is involved, which aligns well with agile practices.

Deployment

Laurie Williams (2018) summarizes her findings from several one-day Continuous Deployment Summits. Her findings are anecdotal but practical. They address security concerns in a rapid and continuous deployment environment common in agile practices. She documents three themes that bolster the deployment of secure software products: communication, culture, and technical practices (Laurie Williams, 2018). Maria, Rodrigues Jr, and Pinto (2015) introduce an accessory to Scrum to improve its security posture. They propose ScrumS processes that run in parallel with the standard Scrum processes. Clark, Collis, Blaze, and Smith (2014) empirically analyze the security consequences of a Rapid Release Cycle (RRC) in the context of the Firefox browser development effort. They find that RRC does not have a negative impact on the quality of secure software developed. Additional detailed work expanding on Laurie Williams (2018) would enhance this area of the SDLC.

PHASE INDEPENDENT

Comprehensive Methodology

The Comprehensive Methodology includes those articles that propose an integration strategy which covers the entire SDLC. We distinguish between those approaches that utilize an existing security framework to guide their integration from those that do not specifically reference an existing security framework. The Comprehensive Methodology category includes articles that do not ground their recommendations on existing security frameworks.

The most frequent method of integration is based on author experience. Authors suggest a diversity of adaptations. Beznosov and Kruchten (2004) suggest utilizing automated methods to mitigate the impact of security practices on agile processes. Ge, Paige, Polack, Chivers, and Brooke (2006) suggest adaptations to Feature Driven Development including tasks for Security Policy Decision and Security Risk Analysis. Security and documentation checklists, tools, training, and a dedicated security review team are suggested by Rajba (2018). Singh (2018) extends XP by adding security-oriented user stories, risk analysis and release inspection.

Keramati and Mirian-Hosseinabadi (2008) as well as Bansal and Jolly (2014) both use a ratings based approach to identify security practices that work well with agile.

Subedi, Alsadoon, Prasad, and Elchouemi (2016) identify limitations of current secure agile approaches. They then propose a secure paradigm for web development based on an agile approach. Siponen et al. (2005) first delineate four requirements that an integration of security practices with agile must fulfill. Then, based on these requirements, they construct security practices in the requirements, design, development and testing phases. Other approaches to adapting agile and secure software development involved literature reviews summarized in guide form (Barbosa & Sampaio, 2015), qualitative interviews which identified the need for security requirements, and the diffusion of security expertise throughout the teams along with the need for security mindfulness (Bartsch, 2011).

Security Framework Integration

The Security Framework Integration section under the Phase Independent category, represents the literature that attempts to integrate secure software development and agile development methods by aligning practices with existing security frameworks. A wide variety of security frameworks have been used in the selected articles. A sampling of

these frameworks includes: Microsoft Secure Development Lifecycle and Microsoft Secure Development Lifecycle for Agile, Cigatel Touchpoints, Common Criteria, COBIT 5 for Risk, Security Engineering – Capability Maturity Model, OWASP Top 10 Security Risks, Building Security In Maturity Model, NIST 800-64, IEC 62443-4-1, and the Comprehensive Lightweight Application Security Process.
Maier et al. (2017) look for common elements across the selected security methodologies. They engage a seasoned Scrum development team to evaluate their approach. They derive several components for their enhanced Scrum methodology including security requirements, trust levels, asset entry points, and security related user stories. From there they move on to Threat Modeling using the Damage potential, Exploitability, Reproducibility, Affected users, Discoverability (DREAD) analysis model developed by Meier et al. (2003). Finally, the authors conclude with risk-based verification involving pair penetration testing, code reviews, documented security controls, static code analysis and a dependency checker.

In a series of research articles (Rindell, Hyrynsalmi, & Leppänen, 2015, 2017; Rindell, Ruohonen, & Hyrynsalmi, 2018), the authors investigate dozens of security practices from numerous security frameworks. In the first paper of the series they conclude that, theoretically, it is feasible to integrate security practices with agile methods (Rindell et al., 2015). The second paper cataloged over 30 security practices and concludes that it is a myth that agile is incompatible with secure software development (Rindell et al., 2017). The authors astutely note that there is a paucity of published empirical support for the success of the integration of security and agile methods. They also optimistically conclude that the myth of incompatibility between agile and secure software development has been busted. This conclusion appears premature. From our perspective, introducing over 30 security practices may cause a significant reduction in agility to the point of questioning the success of the integration.

Wäyrynen, Bodén, and Boström (2004) conclude that an analysis of XP using the Security Engineering – Capability Maturity Model and Common Criteria reveals there are numerous security shortcomings with XP from a security perspective. In order to bolster XP as a secure approach and yet not lose its agility they propose four security enhancements: include a security engineer, document pair programming with the security engineer, document the security architecture, and perform static verification.

Baca and Carlsson (2011) use survey results to determine which security activities from Microsoft SDL, Cigatel Touchpoints, and Common Criteria provide the most benefit with the least impact to agility. Dorca, Munteanu, Popescu, Chioreanu, and Peleskei (2016) focus on a security team providing services to the software development group. Moyon, Beckers, Klepper, Lachberger, and Bruegge (2018) describe an integration of the IEC 62443-4-1, the security for industrial automation and control systems standard, into a Scrum process, specifically the Scaled Agile Framework. Sonia and Banati (2014) propose not only specific security practices for agile processes, they also formulate an agility metric to rate the agility of each Comprehensive Lightweight Application Security Process practice. We also discovered an interesting slide set that seems to describe the genesis of Microsoft's Software Development Lifecycle for Agile (Sullivan, 2010).

Training

Training surfaces as a component for addressing security issues in an agile environment (Poller, Kocksch, Türpe, Epp, & Kinder-Kurlanda, 2017). Oyetoyan, Cruzes, and Jaatun (2016) share results from an empirical, survey-based study between two organizations on some key agile security elements. One of their significant findings is the desire and need for secure software design training. They find that security skill drives security practice usage and that years of development experience correlates with security experience, implying that those new to development haven't had significant training on secure software development. Rajba (2018) also identifies security training as a mitigation to agile security challenges.

Systematic Literature Reviews

Our selected articles include four systematic literature reviews. We include Systematic Literature Reviews as a portion of the 'Phase Independent' category because we found these articles noteworthy and they did not relate directly to a particular phase. Oueslati, Rahman, and ben Othmane (2015) submit a systematic literature review on the challenges of integrating security practices and agile methods. They identify 14 challenges that are unique to integrating security within agile methods and discuss the causes of these challenges. In their systematic literature review, Shuaibu et al. (2015) review 27 development models in regard to their practices. The authors determine that there is no one dominant

model in use in the literature at that time. They do, however, note a consistent use of Threat Modeling as an approach for addressing secure software development. Villamizar et al. (2018) suggest that the understanding of how to handle security within agile is still vague. Their research analyzed publications of approaches that address security requirements in agile software development projects. They outline 21 studies from 2005-2017 and found that these approaches typically modify agile methods by introducing new artifacts or guidelines to handle security issues. Other types of solutions include proposing new conceptional frameworks or providing tool support. Villamizar et al. (2018) suggest that more effort needs to be invested into analyzing existing approaches to mitigate limitations. They found a lack of empirical evaluation research. The paper found that most approaches related to the Scrum method and there were limitations related to environment, people, effort, and resources. Their mapping study indicated several promising avenues for future research. Rindell et al. (2017) perform a systematic literature review to understand the state of integration of security practices and agile methods. They identify 38 security practices, many originating from a variety of security frameworks. Thirty-four of these practices appeared in more than one case in the papers reviewed. The activities aligned well with the Microsoft SDL. As noted elsewhere in this paper, we disagree with the conclusion of Rindell et al. (2017), that the difficulty of integrating security and agile practices has been reconciled. Listing over 30 security practices alongside agile practices raises questions as to the agility of the final result. We believe empirical evidence is needed to substantiate the claim that the integration is successful.

Other

This category is a catch-all for articles that didn't fit well in any of our other categories. Alnatheer, Gravell, and Argles (2010) provide a brief poster as a preliminary work to Alnatheer (2014) dissertation work. Here the author gives an excellent overview of the then current literature. He identifies a number of security related issues for agile development. Through the integration of a systematic literature review and interviews with practitioners, the author addresses many of the agile security issues. His top findings were that the best, albeit expensive, approach is to include at least one security engineer on the project. His second most effective finding was to instill a security mindset on the team. Adelyar (2015) reviews the literature to derive some key secure software develop activities. The author then uses an interview-based case study approach. His analysis yields five security challenge themes emerging from the interview data in relation to integrating security practices in an agile environment. van der Heijden, Broasca, and Serebrenik (2018) use an empirical, survey-based study to identify challenges for developing secure software in a large-scale agile environment. They record three key hurdles for agile at scale. First, they discuss the difficulty of coordinating security objectives in a distributed context. Next, they see problems in having a common mindset with regard to security roles and responsibilities in large-scale efforts. Finally, they mention the challenge of integrating efficient and effective security testing tools into the project.

SUMMARY

Practitioners can use this organization of the literature to find relevant security practices according to their particular needs. It also highlights the tension between agility and security due to their differing objectives. Practitioners must be aware that by simply adopting the latest secure software development practices, they may be jeopardizing the productivity and responsiveness intended by using agile methods.

There is a significant body of literature on the topic of integrating secure software development with agile software development methods. Just like there is no single definitive agile method, there is no single approach to integrate secure development practices with agile. We introduce a novel taxonomy for organizing the knowledge on this topic. We provide two high-level categories: Phase Focused and Phase Independent. Within the Phase Focused category, we see literature dispersed along Software Development Lifecycle phases, with the majority of research in the area of requirements. In the Phase Independent category, there are two main streams. First, Comprehensive Methodologies provide literature and experience-based guidance for integrating agile with secure practices. Second, we formulate a segment identified as Security Framework Integration, composed of research based on existing security frameworks integrated with agile approaches. We also recognize training, existing Systematic Literature Reviews and a general category named Other.

One of our concerns with the Phase Focused area is the very alignment of activities with discrete SDLC phases. Agile is not a series of mini-waterfall activities. The integration of security practices along the lines of traditional SDLC

phases seems like a mis-aligned approach to agile software development. A better approach might be to take an agile methodology like Scrum with its' artifacts and processes and inject security practices and philosophies into that native agile environment while remaining true to the agile values and principles.

In the Phase Independent category, the concern is adopting every suggested practice from security frameworks and encumbering software development to the point that it is no longer recognizable as an agile methodology. Quantitative impact in terms of feature output might be considered as an evaluation criterion to augment the qualitative, survey-based feedback that has been used in much of the research to-date.

Although there is some empirical research, there is an opportunity to extend the body of knowledge in this area. The literature shows that there are widely varying tolerances for the integration of heavyweight security practices into agile methods. Some of the published research used formulas and expert assessment to try to quantify a particular security practice's impact on agility. Each team will need to assess their own risk tolerance and their own desire for agility. Balancing these two requirements–security and agility–is still an ongoing discussion, requires more research, and will require each organization to make their own decisions.

REFERENCES

Adelyar, S. H. (2015). *Towards Secure Agile Agent-Oriented System Design.* Paper presented at the Cloud Engineering (IC2E), 2015 IEEE International Conference on.

Adelyar, S. H., & Norta, A. (2016). *Towards a Secure Agile Software Development Process.* Paper presented at the 2016 10th International Conference on the Quality of Information and Communications Technology (QUATIC).

Alnatheer, A. (2014). *The investigation of security issues in agile methodologies.* University of Southampton,

Alnatheer, A., Gravell, A. M., & Argles, D. (2010). *Agile security issues: an empirical study.* Paper presented at the Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement.

Arbain, A. F. B., Ghani, I., & Kadir, W. M. N. W. (2014). *Agile non functional requiremnents (NFR) traceability metamodel.* Paper presented at the Software Engineering Conference (MySEC), 2014 8th Malaysian.

Avizienis, A., Laprie, J.-C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing, 1*(1), 11-33.

Azham, Z., Ghani, I., & Ithnin, N. (2011). *Security backlog in Scrum security practices.* Paper presented at the Software Engineering (MySEC), 2011 5th Malaysian Conference in.

Baca, D., & Carlsson, B. (2011). *Agile development with security engineering activities.* Paper presented at the Proceedings of the 2011 International Conference on Software and Systems Process.

Balijepally, V. G., Mahapatra, R. K., & Nerur, S. P. (2006). Assessing personality profiles of software developers in agile development teams. *Communications of the Association for Information Systems, 18*(1), 4.

Bansal, S. K., & Jolly, A. (2014). *An encyclopedic approach for realization of security activities with agile methodologies.* Paper presented at the Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference-.

Barbosa, D. A., & Sampaio, S. (2015). *Guide to the Support for the Enhancement of Security Measures in Agile Projects.* Paper presented at the Agile Methods (WBMA), 2015 6th Brazilian Workshop on.

Bartsch, S. (2011). *Practitioners' perspectives on security in agile development.* Paper presented at the Availability, Reliability and Security (ARES), 2011 Sixth International Conference on.

Beck, K., et al. (2001). Manifesto for Agile Software Development. Retrieved from http://www.agilealliance.org

Beznosov, K., & Kruchten, P. (2004). *Towards agile security assurance.* Paper presented at the Proceedings of the 2004 workshop on New security paradigms.

Bowen, J. P., Hinchey, M., Janicke, H., Ward, M., & Zedan, H. (2018). Formality, Agility, Security, and Evolution in Software Engineering. *IEEE Computer Society, 47(10),* 86-89.

Brady, K. (2006). AGILE/SCRUM Fails to get to grips with Human Psychology. In: Hämtad.

Chivers, H., Paige, R. F., & Ge, X. (2005). *Agile security using an incremental security architecture.* Paper presented at the International Conference on Extreme Programming and Agile Processes in Software Engineering.

Chóliz, J., Vilas, J., & Moreira, J. (2015). *Independent security testing on agile software development: a case study in a software company.* Paper presented at the Availability, Reliability and Security (ARES), 2015 10th International Conference on.

Clark, S., Collis, M., Blaze, M., & Smith, J. M. (2014). *Moving targets: Security and rapid-release in firefox.* Paper presented at the Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security.

Conboy, K. (2009). Agility from first principles: reconstructing the concept of agility in information systems development. *Information Systems Research, 20*(3), 329-354.

Corbin, J. M., & Strauss, A. (1990). Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative sociology, 13*(1), 3-21.

Daneva, M., & Wang, C. (2018). *Security Requirements Engineering in the Agile Era: How Does it Work in Practice?* Paper presented at the 2018 IEEE 1st International Workshop on Quality Requirements in Agile Projects (QuaRAP).

Dorca, V., Munteanu, R., Popescu, S., Chioreanu, A., & Peleskei, C. (2016). *Agile approach with Kanban in information security risk management.* Paper presented at the Automation, Quality and Testing, Robotics (AQTR), 2016 IEEE International Conference on.

Farid, W. M. (2012). *The Normap methodology: Lightweight engineering of non-functional requirements for agile processes.* Paper presented at the Software Engineering Conference (APSEC), 2012 19th Asia-Pacific.

Galvez, R., & Gurses, S. (2018). *The Odyssey: modeling privacy threats in a brave new world.* Paper presented at the 2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW).

Ge, X., Paige, R. F., Polack, F., & Brooke, P. (2007). *Extreme programming security practices.* Paper presented at the International Conference on Extreme Programming and Agile Processes in Software Engineering.

Ge, X., Paige, R. F., Polack, F. A., Chivers, H., & Brooke, P. J. (2006). *Agile development of secure web applications.* Paper presented at the Proceedings of the 6th international conference on Web engineering.

Ghani, I., Azham, Z., & Jeong, S. R. (2014). Integrating Software Security into Agile-Scrum Method. *KSII Transactions on Internet & Information Systems, 8*(2).

Hassan, M., Mubashir, M., Shabir, M., & Ullay, M. (2018). Software Quality Assurance Techniques: A Review. *International Journal of Information, Business and Management, 10*(4), 214-221.

Keramati, H., & Mirian-Hosseinabadi, S.-H. (2008). *Integrating software development security activities with agile methodologies.* Paper presented at the Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on.

Kitchenham, B. J. K., UK, Keele University. (2004). Procedures for performing systematic reviews. *33*(2004), 1-26.

Knauss, E., Liebel, G., Schneider, K., Horkoff, J., & Kasauli, R. (2017). *Quality Requirements in Agile as a Knowledge Management Problem: More than Just-in-Time.* Paper presented at the 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW).

Kongsli, V. (2006). *Towards agile security in web applications.* Paper presented at the Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications.

Kongsli, V. (2007). *Security testing with Selenium.* Paper presented at the Companion to the 22nd ACM SIGPLAN conference on Object-oriented programming systems and applications companion.

Maier, P., Ma, Z., & Bloem, R. (2017). *Towards a Secure SCRUM Process for Agile Web Application Development.* Paper presented at the Proceedings of the 12th International Conference on Availability, Reliability and Security.

Maria, R. E., Rodrigues Jr, L. A., & Pinto, N. A. (2015). *ScrumS: a model for safe agile development.* Paper presented at the Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems.

Mead, N. R., Viswanathan, V., & Padmanabhan, D. (2008). *Incorporating security requirements engineering into the dynamic systems development method.* Paper presented at the Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International.

Meier, J., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., & Murukan, A. J. M. C. (2003). Improving web application security: threats and countermeasures. *3*.

Moyon, F., Beckers, K., Klepper, S., Lachberger, P., & Bruegge, B. (2018). *Towards continuous security compliance in agile software development at scale.* Paper presented at the 2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE).

NIST. (2019). National Vulnerability Database. Retrieved from https://nvd.nist.gov/vuln/search/statistics?adv_search=false&form_type=basic&results_type=statistics&search_type=all

Othmane, L., Angin, P., Weffers, H., & Bhargava, B. (2014). Extending the agile development process to develop acceptably secure software. *IEEE Transactions on Dependable and Secure Computing, 11*(6), 497-509.

Othmane, L. B., & Ali, A. (2016). *Towards Effective Security Assurance for Incremental Software Development the Case of Zen Cart Application.* Paper presented at the Availability, Reliability and Security (ARES), 2016 11th International Conference on.

Oueslati, H., Rahman, M. M., & ben Othmane, L. (2015). *Literature review of the challenges of developing secure software using the agile approach.* Paper presented at the Availability, Reliability and Security (ARES), 2015 10th International Conference on.

OWASP. (2018). OWASP Secure Software Development Lifecycle Project. Retrieved from https://www.owasp.org/index.php/OWASP_Secure_Software_Development_Lifecycle_Project

Oyetoyan, T. D., Cruzes, D. S., & Jaatun, M. G. (2016). *An empirical study on the relationship between software security skills, usage and training needs in agile settings.* Paper presented at the Availability, Reliability and Security (ARES), 2016 11th International Conference on.

Page, V., Dixon, M., & Choudhury, I. (2007). Security risk mitigation for information systems. *BT technology journal, 25*(1), 118-127.

Peeters, J. (2005). *Agile security requirements engineering.* Paper presented at the Symposium on Requirements Engineering for Information Security.

Poller, A., Kocksch, L., Türpe, S., Epp, F. A., & Kinder-Kurlanda, K. (2017). *Can Security Become a Routine?: A Study of Organizational Change in an Agile Software Development Group.* Paper presented at the CSCW.

Rajba, P. (2018). *Challenges and mitigation approaches for getting secured applications in an enterprise company.* Paper presented at the Proceedings of the 13th International Conference on Availability, Reliability and Security.

Raschke, W., Zilli, M., Baumgartner, P., Loinig, J., Steger, C., & Kreiner, C. (2014). *Supporting evolving security models for an agile security evaluation.* Paper presented at the Evolving Security and Privacy Requirements Engineering (ESPRE), 2014 IEEE 1st Workshop on.

Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2015). *A comparison of security assurance support of agile software development methods.* Paper presented at the Proceedings of the 16th International Conference on Computer Systems and Technologies.

Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2017). *Busting a myth: Review of agile security engineering methods.* Paper presented at the Proceedings of the 12th International Conference on Availability, Reliability and Security.

Rindell, K., Ruohonen, J., & Hyrynsalmi, S. (2018). *Surveying Secure Software Development Practices in Finland.* Paper presented at the Proceedings of the 13th International Conference on Availability, Reliability and Security.

Sachdeva, V., & Chung, L. (2017). *Handling non-functional requirements for big data and IOT projects in Scrum.* Paper presented at the Cloud Computing, Data Science & Engineering-Confluence, 2017 7th International Conference on.

Shuaibu, B. M., Norwawi, N. M., Selamat, M. H., & Al-Alwani, A. (2015). Systematic review of web application security development model. *Artificial Intelligence Review, 43*(2), 259-276.

Singh, A. (2018). *Integrating the Extreme Programing Model with Secure Process for Requirement Selection.* Paper presented at the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA).

Siponen, M., Baskerville, R., & Kuivalainen, T. (2005). *Integrating security into agile development methods.* Paper presented at the System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on.

Sonia, A. S., & Banati, H. (2014). FISA-XP: An Agile-based Integration of Security Activities with Extreme Programming. SIGSOFT So w. Eng. *Notes, 39*(3), 1-14.

Subedi, B., Alsadoon, A., Prasad, P., & Elchouemi, A. (2016). *Secure paradigm for web application development.* Paper presented at the RoEduNet Conference: Networking in Education and Research, 2016 15th.

Sullivan, B. (2010). Agile Security; or, How to defend applications with five-day release cycles. Retrieved from https://www.blackhat.com/presentations/bh-dc-10/Sullivan_Bryan/BlackHat-DC-2010-Sullivan-SDL-Agile-slides.pdf

Tappenden, A., Beatty, P., Miller, J., Geras, A., & Smith, M. (2005). *Agile security testing of web-based systems via httpunit.* Paper presented at the Agile Conference, 2005. Proceedings.

Terpstra, E., Daneva, M., & Wang, C. (2017). *Agile Practitioners' Understanding of Security Requirements: Insights from a Grounded Theory Analysis.* Paper presented at the 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW).

Tondel, I. A., Jaatun, M. G., & Meland, P. H. (2008). Security requirements for the rest of us: A survey. *IEEE software, 25*(1).

van der Heijden, A., Broasca, C., & Serebrenik, A. (2018). *An empirical perspective on security challenges in large-scale agile software development.* Paper presented at the Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement.

VersionOne. (2018). 12th Annual State of Agile Report.

Villamizar, H., Kalinowski, M., Viana, M., & Fernández, D. M. J. a. p. a. (2018). A Systematic Mapping Study on Security in Agile Requirements Engineering.

Wang, W., Gupta, A., & Niu, N. (2018). *Mining Security Requirements from Common Vulnerabilities and Exposures for Agile Projects.* Paper presented at the 2018 IEEE 1st International Workshop on Quality Requirements in Agile Projects (QuaRAP).

Wäyrynen, J., Bodén, M., & Boström, G. (2004). *Security engineering and eXtreme programming: An impossible marriage?* Paper presented at the Conference on Extreme Programming and Agile Methods.

Williams, L. (2012). What agile teams think of agile principles. *Communications of the ACM, 55*(4), 71-76.

Williams, L. (2018). *Continuously integrating security.* Paper presented at the Proceedings of the 1st International Workshop on Security Awareness from Design to Deployment.

Yu, W. D., & Le, K. (2012). *Towards a secure software development lifecycle with square+ r.* Paper presented at the Computer Software and Applications Conference Workshops (COMPSACW), 2012 IEEE 36th Annual.