Information and Computer Security, https://doi.org/10.1108/ICS-12-2018-0138

Collaborative Security Risk Estimation in Agile Software Development

Inger Anne Tøndel (NTNU/SINTEF Digital), Martin Gilje Jaatun (SINTEF Digital), Daniela Soares Cruzes (SINTEF Digital), Laurie Williams (NCSU)

Abstract
Purpose: Today, agile software development teams in general do not adopt security risk assessment practices in an ongoing manner to prioritize security work. Protection Poker is a collaborative and lightweight software security risk estimation technique that is particularly suited for agile teams. Motivated by a desire to understand why security risk assessments has not yet gained widespread adoption in agile development, this study assess to what extent the Protection Poker game would be accepted by agile teams, and how it can be successfully integrated into the agile practices.
Design/methodology/approach: Protection Poker was studied in capstone projects, in teams doing a graduate software security course, and in sessions with industry representatives. Data was collected via questionnaires, observations, and group interviews.
Findings: Results show that Protection Poker has the potential to be adopted by agile teams. Key benefits include good discussions on security and the development project, and increased knowledge and awareness. Challenges include ensuring efficient use of time, and gaining impact on the end-product.
Research limitations/implications: Using students allowed for easy access to subjects and an ability to collect rich data over time, but at the cost of generalisability to professional settings. Results from interactions with professionals supplement the data from students, showing similarities but also differences in their opinion on Protection Poker.
Originality/value: The paper proposes ways to tackle the main obstacles to adoption of the Protection Poker technique, as identified in this study.

Keywords: software security, risk assessments, agile development, case study, secure software engineering, Protection Poker

1 Introduction

Agile development methods have gained widespread adoption in the software industry, and agile methods are now used for all types of software development and for various types of systems, including very large development projects (Dingsøyr et al., 2018). Current evidence shows that security work is often neglected in agile projects (Oueslati et al., 2015, Terpstra et al., 2017, Khaim et al., 2016, Tøndel et al., 2017), and that teams generally do not estimate security risks in an ongoing manner to inform the security requirements work (Tøndel et al., 2017). Risk management is important for making decisions on security activities in agile development, since full security analysis in every sprint is not possible (Oueslati et al., 2015).

Risk management activities both within cyber security and software security are motivated by similar goals: to ensure that security activities are in line with organisational goals and

objectives; and to address the security needs in an effective and timely manner (ISO/IEC, 2011, Caralli et al., 2007, NIST, 2010). For software security, risk management involves effectively and cost-efficiently identifying security vulnerabilities and risks and prioritizing mitigations (Oueslati et al., 2015); and using risk as a basis for prioritizing development efforts and make trade-off decisions (McGraw, 2006, Chandra, 2008). Additionally, through raising awareness, teams gain an improved understanding of what factors may lead to negative outcomes (Chandra, 2008) and become more able to think like an attacker (McGraw et al., 2016). Achieving these goals is, however, not a straight-forward task. Understanding and assessing security risk is known to be a complex challenge, requiring a large skill set (Tøndel et al., 2017).

Limited empirical data is available on what makes risk management difficult, both within cyber security and software security. A review of risk analysis methods for IT systems (Sulaman et al., 2013) identified a lack of evaluation of risk analysis methods. Despite the mantra that all security work should be risk based, a study among information security professionals (Jourdan et al., 2010) unveiled that as many as 25 % stated that risk analysis was never or rarely performed for their department or organisation. A main challenge is the estimation of likelihood and cost, in part due to limited availability of historical data and constantly changing risk factors (Fenz and Ekelhart, 2010, Cybenko, 2006, Gerber and Von Solms, 2005, Rhee et al., 2012, Tøndel et al., 2015).

Few research papers report on risk analysis methods specifically tailored towards agile teams. Protection Poker (Williams et al., 2010) is a notable exception. Protection Poker is based on the Planning Poker game (Grenning, 2002) that is used for effort estimation in agile projects. Protection Poker is intended to be played as part of every iteration planning meeting, in order to rank the security risk of each feature to be implemented in that iteration, and to identify security mechanisms that should be implemented to maintain an acceptable risk level. The full team together identifies assets related to the features and uses the Protection Poker game to rank the features according to their security risk; assessing the value of their assets and the ease of attack. Although proposed by Williams et al. (2010) in 2010, Protection Poker is still not widely used in the software industry.

Protection Poker is a promising technique to study further due to its potential to increase security awareness and knowledge in the full development team (Williams et al., 2010). Additionally, previous studies have identified security benefits that can be traced back to using an incremental risk analysis approach (Baca et al., 2015a), and have identified the need for more research on practical ways of doing risk analysis in an agile context, i.e. lightweight and continuously throughout the project (Tøndel et al., 2017). By studying adoption of Protection Poker and how Protection Poker can be successfully integrated into the practices of agile development teams, our aim is to build knowledge on how to increase adoption of security risk assessment practices by agile teams. As Protection Poker has not yet gained widespread adoption, understanding potential reasons why this is the case can additionally help improve Protection Poker, and other techniques with similar goals as Protection Poker.

This paper presents a family of studies of applying Protection Poker in three different settings: by six capstone development project teams; by 16 teams in a graduate software security course; and in sessions with industry practitioners. Our investigation was centred on the following research questions:

- RQ1: To what extent is Protection Poker accepted by the players, both in the short term and in the longer term?

- RQ2: What lessons learned and improvements to Protection Poker are identified by the players?

This paper is an extended version of Tøndel et al. (2018) that presented the results of using Protection Poker in the capstone development projects. Compared to the previous paper, this paper adds results from sessions of applying Protection Poker with industry representatives and in a graduate-level software security course. Additionally, this paper provides a more thorough overview of literature on security risk assessment in agile development and on challenges to adoption of security activities in agile development. Furthermore, this paper offers more concrete advice on when and how to adopt Protection Poker in agile development projects.
The remainder of this paper is organised as follows. Section 2 gives an overview of relevant literature on adoption of risk assessment practices in agile software development. Section 3 gives a more thorough introduction to Protection Poker. Section 4 explains the research method used in the study. Section 5 presents the results of the study, and Section 6 discusses the implications of these results. Section 7 discusses threats to validity. Section 8 concludes the paper.

2 Adoption of security risk assessment practices by agile teams

This section gives an overview of the current state of software security risk assessment in agile software development, and introduces literature on challenges and factors important for adoption of security activities in agile development.

2.1 Approaches to security risk assessment in agile development

Software development projects need to deal with many types of risks, including security risks. In agile development practices, risk management can be said to be treated implicitly (Tavares et al., 2017, Odzaly et al., 2017), and the guidance provided by agile methods when it comes to risk management is *"very general"* (Nyfjord and Kajko-Mattsson, 2008). Few research papers provide concrete guidance on how to tackle security risk management for agile projects (Tøndel et al., 2017). The most notable technique available is Protection Poker. Additionally, the security enhanced agile software development process (SEAP) studied at Ericsson (Baca et al., 2015b) provides high-level suggestions for performing incremental risk analysis, in addition to other practices such as adding more security resources to the teams and performing security activities such as code review and penetration testing. The results of a study of SEAP reported improved identification and handling of risk. Thus, risk management was found to be more cost-efficient with SEAP than with the approach previously used by Ericsson, because security issues were dealt with in a more distributed fashion with more issues solved directly by the team. The details of how risk analysis and risk management was conducted with SEAP is not available, however the frequency of risk analysis was increased, the scope for each analysis was reduced, and the approach was more distributed.

Within the area of cyber security, standards, guidelines and research papers suggest different ways of managing risk and performing risk assessments. Some of the major ones are ISO/IEC 27005 (ISO/IEC, 2011), OCTAVE Allegro (Caralli et al., 2007), and the NIST Risk Management Framework (RMF) (NIST, 2010). These documents suggest practices that concern assessing the risk, making decisions on how to treat the risk, follow up on these decisions, and communicating information related to security risks in the organisation (Tøndel et al., 2017).

Within software security, risk assessment practices are commonly included in software security frameworks, maturity models and Security Development Lifecycles (SDLs). The OWASP

Software Assurance Maturity Model (OpenSAMM) (Deleersnyder et al., 2017) include activities on performing threat assessments. The seven touchpoints for software security (McGraw, 2004) include the touchpoints abuse cases and risk analysis. Microsoft SDL (Howard and Lipner, 2006) includes activities to perform security and privacy risk assessments, attack surface analysis/reduction and perform threat modelling. One of the main sources for information about software security practices is the Building Security In Maturity Model (BSIMM) (Williams et al., 2018), mainly giving an overview of practices of big companies. Though BSIMM does not have an activity that is named "risk analysis", it contains several activities related to such an activity, e.g., *"Use a risk questionnaire to rank applications"* which has an adoption of 45 % in the 2017 version of BSIMM, and *"Require security sign-off"* which concern a process for risk acceptance and has an adoption of 30 %. Regarding smaller companies, we are only aware of one study in this respect (Tøndel et al., 2017), finding that risk assessment practices in public development organisations were not based on risk analysis, but rather driven by compliance. The organisations performed risk analysis on some level, but the practices were by and large not integrated with and considered to be especially relevant for development. Thus, more empirical studies are needed on what can be done to increase adoption of security risk assessment by software development projects.

Table 1 provides an overview of the risk management approaches mentioned above, and their usefulness for agile development teams. Note that a complete overview of available methods is outside the scope of this paper.

Table 1: A selection of existing approaches and their agile usefulness

Approach and references	**Risk management activities included**	**Usefulness for agile teams**
Agile methodologies concerning security risks		
SEAP (Baca et al., 2015b)	High-level suggestions for risk analysis.	Details are not available.
Risk analysis methodologies		
ISO/IEC 27005 (ISO/IEC, 2011)	Information security risk management. Organisational approach.	Not directly applicable to agile software development.
OCTAVE Allegro (Caralli et al., 2007)	Information security risk management. Organisational approach.	Not directly applicable to agile software development.
NIST Risk Management Framework (RMF) (NIST, 2010)	Integration of risk management into software development.	Describes a continuous process, but may be too comprehensive for many agile projects.
Software security maturity models		
OpenSAMM (Deleersnyder et al., 2017)	Contains activities for threat assessment, security requirements and more.	Activities can be adopted by agile teams. High-level descriptions.
BSIMM [16](Williams et al., 2018)	Based on studies of predominantly large companies.	Activities can be adopted by agile teams. High-level descriptions.

	Contains some risk-related activities, e.g. attack models.	
Secure Software Development Lifecycles		
Touchpoints for software security (McGraw, 2004)	Includes abuse cases and risk analysis.	Activities can be adopted by agile teams. High-level descriptions.
Microsoft SDL (Howard and Lipner, 2006)	Includes security and privacy risk assessments, attack surface analysis/reduction, threat modelling.	Agile version of this SDL is available (Microsoft, 2012) that explains how to integrate these practices in agile development.

2.2 Challenges of security activities in agile development

Research in software security covers a varied range of approaches and processes that deal with security during software development. Several approaches have been suggested to incorporate security into the software development lifecycle (Oueslati et al., 2015). When aiming to apply security practices in agile software development it is essential to take into consideration the agile principles of *"Individuals and interactions over processes and tools", "Working software over comprehensive documentation", "Customer collaboration over contract negotiation",* and *"Responding to change over following a plan"* (Beck et al., 2001). Current studies have identified challenges when security work is expected to be guided by these same principles (see Table 2 and Table 3 for an overview of the cited studies).

Oueslati et al. (2015) identified a set of 14 challenges of developing secure software using the agile development approach and methods reported in the literature. The challenges were categorized as *"Software development life-cycle challenges"*, *"Incremental development challenges"*, *"Security assurance challenges"* and *"Awareness and collaboration challenges"*. Several of the challenges relate to the need to fit security activities into the short iteration times, the need to deal with changing functional requirements that may break previous security analysis and decisions, and the reliance on documentation that is common within security work.

Cruzes et al. (2018) identified twenty-one challenges to threat modelling in agile development based on a case study in a development organisation. The principle of *"Individuals and interactions over processes and tools"* was challenging especially because it was hard to get effective meetings with clear and actionable outputs. The principle of *"Working software over comprehensive documentation"* is many times misunderstood by agile teams to be “no documentation” and especially developers have lost the focus on documenting their work, or understanding the need for documentation. Security work is many times very much based on the documentation of the decisions, risks and assets. This study gives us motivation to investigate further how to make the security discussion meetings more effective.

Türpe and Poller (2017) theorize about tensions between the characteristics of security requirements and security work on the one hand, and the way Scrum manages development work on the other. The authors find three different ways of managing security work: as bug fixing on demand, continuously as a quality requirement through the definition of “done,” or as prioritized and planned development work through the product backlog. All of them are found inadequate. On-demand fixing rarely leads to substantial security improvement. As a quality requirement, security has a complex relationship with development work and is difficult

to verify. Security features in the backlog would be a suitable approach to many security concerns, but they compete with other requirements and may also need special expertise to design and implement effectively.

Terpstra et al. (2017) performed a study of how practitioners reason about and cope with security requirements in agile development, based on postings on LinkedIn. They identified 21 challenges and 15 coping strategies, and used these to create a conceptual model. Challenges pointed to in this model is a limited business case for security, unclear ownership of security requirements, limited organisational effort to educate developers on security, limited incentives to care about security, and the varying perceptions of priority among business representatives. The agile principle of *"Individuals and interactions over processes and tools"* lead to the priorities of security work being highly reliant on the people involved in the project. Both Alsaquaf et al. (2017) in a literature review of quality requirements work in agile development, and Terpstra et al. (2017) identify the product owner as a hindrance for quality requirements being properly addressed, as the product owner commonly have a *"heavy workload"* and *"insufficient availability"* in addition to a *"lack of knowledge"* on the quality aspects (Alsaqaf et al., 2017). Challenges on prioritizing time and money on security are however not specific for agile development. Kanniah and Mahrin (2016) has previously in a review of 44 primary studies identified *"Adequate Development Time"* and *"Adequate Budget/Cost"* as commonly cited factors that impact the successful implementation of secure software development practices. Additionally, Geer (2010) found in a survey that *"Too time consuming"* and *"Requires too many resources"* was the main self-reported reasons for not adopting secure SDLs, together with not being aware of the methodologies.

Table 2: Selected challenges identified in the literature

Author and reference	**Topic**	**Challenges**
Oueslati et al. (2015)	Software security in agile development (literature review)	*"Software development life-cycle challenges"* (security activities not included; hard to integrate security in every iteration due to short iteration times) *"Incremental development challenges"* (dealing with changes) *"Security assurance challenges"* (documentation; testing; unstable development process) *"Awareness and collaboration challenges"* (security requirements neglected; lack of experience and security awareness; separate the developer and reviewer roles) *"Security management challenges"* (giving priority to security).
Cruzes et al. (2018)	Threat modelling in agile (case study)	*"Asset Identification"* (documentation) *"Data Flow Diagrams"* (documentation; level of abstraction; interfaces; link to code; maintainability) *"Modeling Meeting"* (effective meetings; who should participate; distributed settings; finding what is good enough; expertise) *"STRIDE"* (communication channel focus)

		"Outputs from the Session" (make it actionable; follow up; prioritizing security; false sense of security)
Türpe and Poller (2017)	Security requirements in SCRUM	Security as bug fixing on demand, continuously as a quality requirement, or as prioritized and planned development work through the backlog
Terpstra et al. (2017)	Security requirements in agile (study of practitioners' posts on LinkedIN)	Unclear business case for security (hard to sell as business value; costly) Unclear ownership of security requirements (forget about security; delivered late) Perceptions of priority (differing priorities; not prioritized by customers and Product Owners) Understanding of security (low awareness among developers; lack of training; dependent on individuals) Organisational context (lack of involvement of security experts; Product Owner become a limiting factor; organisational structure can make or break modifications to requirements) Poorly defined security requirements
Alsaquaf et al. (2017)	Quality requirements in agile (literature review)	Technique (no widely accepted technique; inadequacy of existing techniques; traceability) Priorities (functionality is prioritized; ignore some types of requirements; validated late; insufficient analysis) Product Owner (lack of knowledge; workload; availability; dependence)

Table 3: Selected factors identified in literature, influencing adoption of software security practices

Author and reference	**Topic**	**Factors**
Kanniah and Mahrin (2016)	Implementation of secure software development practices (literature review)	*"Institutional Context"* (change management; policy enforcement; training; incentives; culture and organisational objectives; security experts) *"People and Action"* (developer and project manager attitude and skills; management support) *"Project Content"* (tool support; budget; security experts/team; development time; development methodology) *"System Development Process"* (methodology; security requirements; security policies/standards/guidelines; metrics/KPI)
Geer (2010)	Adoption of secure SDLs (online survey)	The three most popular responses were: *"Too time consuming"* *"Not aware of methodologies"* *"Requires too many resources"*

3 Protection Poker

Protection Poker is a technique that is designed to engage the whole team in discussing security problems, and does so in a way that is concrete and (hopefully) fun. Additionally, Protection Poker is specifically designed to be applied for each development iteration, ensuring that security is considered throughout. Protection Poker results in a ranking of features based upon their security risk. Protection Poker was originally proposed by Williams et al. (Williams et al., 2010, Williams et al., 2009), and later modified by Jaatun and Tøndel (Jaatun and Tøndel, 2016). As can be seen from Table 4, these two variations of Protection Poker differ on two aspects; the risk calculation and the card values used.

Table 4. Overview of the two Protection Poker variations

	Original Protection Poker (Williams et al., 2010)	***Modified Protection Poker*** (Jaatun and Tøndel, 2016)
Risk calculation	risk = ∑(asset values) x (ease of attack)	risk = ∑(asset values) x (exposure)
Card values	1, 2, 3, 5, 8, 13, 20, 40, 100	<10, 20, 30, 40, 50, 60, 70, 80, 90, 100
Used in study	Graduate projects	Capstone projects, industry events

Protection Poker is designed to be played during an iteration planning meeting with the participation of the full development team. One person should have the role as moderator, and this person will be responsible for leading the team through the game and pointing the discussions in a good direction. Ideally, a separate person should be tasked with taking notes on important security solutions and ideas that emerge during play. Focus is on the specific requirements the team will likely implement during the next iteration. A basic overview of the steps involved in playing Protection Poker can be found in Figure 1. The actual playing using the Protection Poker cards is done in steps 4 and 5. Players use the cards to make votes on the risk involved in the requirement they are playing on, and the votes are a basis for further discussions on the risk, and eventually agreeing on a risk value for the requirement. This agreement may require several rounds of voting by using the Protection Poker cards. Below we explain two central concepts of the game, namely risk calculation and calibration.

Risk is always related to a requirement that is to be implemented in the next iteration; which is often new, enhanced or corrected functionality. Exposure/ease of attack relates to how hard or easy the added or changed functionality makes it to attack the system. For asset value, one identifies the assets that are related to a requirement and considers their value for various actor types. Assets are typically considered to be *"data stored in database tables or system processes that the new functionality controls"* (Williams et al., 2010), however in this study we did not use a strict definition of the term asset. In previous work (Jaatun and Tøndel, 2008), we have defined assets as *"anything of value that needs to be protected"*.

To be able to prioritize between requirements and to avoid that high-risk projects assign every requirement a high risk value, the numbers assigned need be spread. Thus, the highest card (100) should be used for asset values and exposures/ease of attack that are high for this project, and similarly the lowest card (<10 or 1)) should be given to asset values and exposures/ease of attack that are low for this project. The goal is not to establish a "perfect" and "universal" risk value, but rather to rank the security risk of the requirements in order to be able to better prioritize security effort. Therefore, a calibration is recommended in the beginning of the playing Projection Poker in order to arrive at a common understanding of the end-points of the

scale, i.e., the team agrees what a '<10'/'1' or a '100' means for this product. When playing about asset value and exposure/ease of attack, numbers should be assigned relative to these endpoints, as well as relative to the values assigned for previously-assessed assets and features.

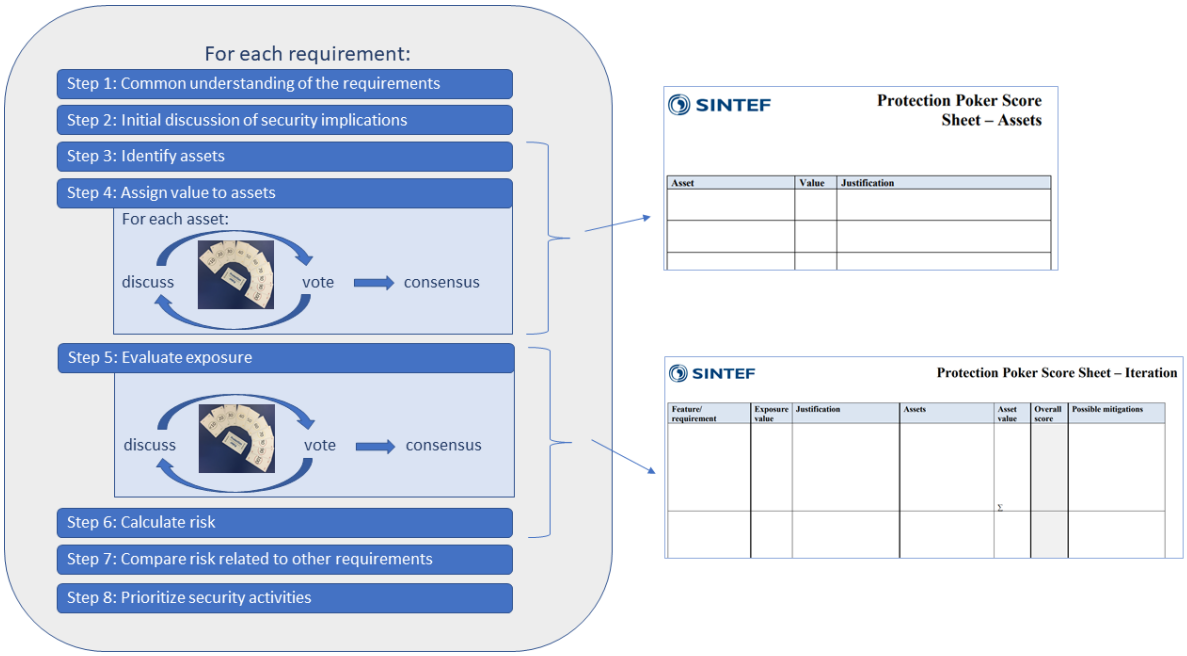

Figure 1. Playing Protection Poker (modified version) (Tøndel et al., 2018)

4 Research Methodology

This section gives an overview of the research method used for this study. The study was performed in three parts:

- a case study of six capstone projects lasting about three months, all using the modified Protection Poker version
- single sessions of playing Protection Poker with industry practitioners, either at company visits or at conferences, all using the modified Protection Poker version
- a case study of 16 project teams in a graduate-level software security class, using the original Protection Poker version

4.1 Capstone projects

The study was performed in the Customer Driven Project course (TDT4290) at the Norwegian University for Science and Technology (NTNU), autumn 2016. This course is mandatory for fourth year computer science students. In this course, the students are divided into development teams (5-8 students per team). Every team is given a development project from an external customer (i.e., private companies, public organisations or research institutes). The students are expected to investigate the needs of the customer, develop software, do some testing of this software and document everything in a report and in a presentation given to the customer. In general, all student groups use agile methodologies to some extent. Six groups, consisting of 34 students in total, were required to use Protection Poker for their project. These groups developed various systems: an app for pupils, a game, an algorithm, a web-based system to register projects, and an isolated system for annotating safety arguments.

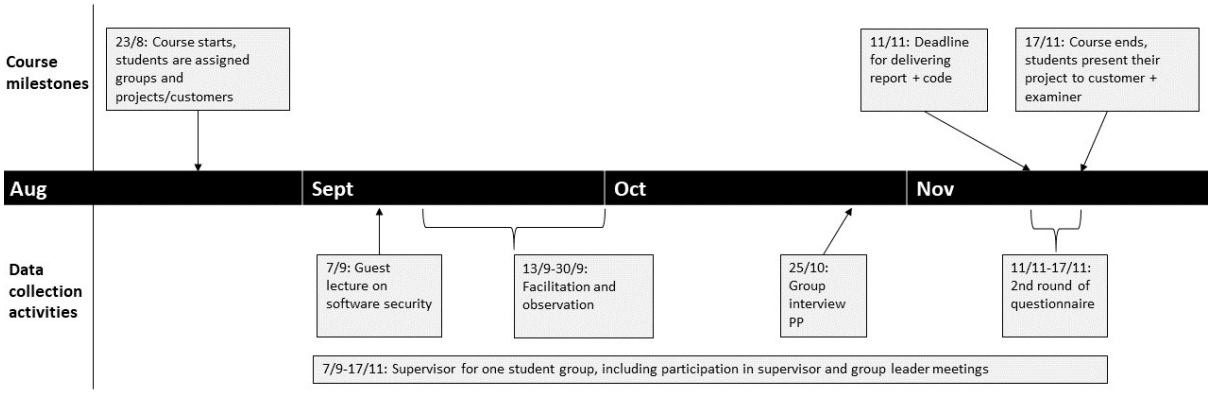

Figure 2. Overview of data collection activities (Tøndel et al., 2018)

An overview of data collection activities can be found in Figure 1. As most students had received limited formal training on software security before this course, we arranged a lecture where all students were given a short plenary introduction to software security and the Protection Poker game. They played the game on an example project and responded to a questionnaire that covered the students' acceptance of the technique. Data collection proceeded through facilitation and observations of students playing Protection Poker in their group, and the observations were followed by group interviews towards the end of the course, allowing detailed student feedback on the technique. Additionally, the main author of this paper acted as supervisor for one of the student groups and took part in project manager and supervisor meetings throughout the course. The questionnaire on acceptance was repeated towards the end of the course. The study has been reported to the national Data Protection Official for research.

The main motivation for using a questionnaire was to capture students' immediate and longer-term acceptance of the Protection Poker technique (RQ1). A questionnaire could easily reach many students and could easily be repeated. We decided to base the questionnaire on the Technology Acceptance Model (TAM) (Davis, 1985) for two reasons. First, TAM, although criticized by some (Li, 2010), is considered a highly influential and commonly employed theory for describing an individual's acceptance of information systems. Thus, we believed TAM could help us understand the different reasons for acceptance of Protection Poker by the students, and that TAM-based questions could trigger comments from the students related to acceptance. Second, we were able to adapt questions from an existing questionnaire (Caroli and Caetano, 2015) to the phenomena we are studying.

The TAM, adapted from the Theory of Reasoned Action (Ajzen and Fishbein, 1980) and originally proposed by Davis, suggests that when users are presented with a new technology, a number of factors influence their decision about how and when they will use it, notably:

- Perceived usefulness: this was defined by Davis as *"the degree to which a person believes that using a particular system would enhance his or her job performance"* (Davis, 1989)
- Perceived ease of use: Davis defined this as *"the degree to which a person believes that using a particular system would be free from effort"* (Davis, 1989)
- External variables: include *"system characteristics, training, user involvement in design, and the nature of the implementation process"* (Venkatesh and Davis, 1996)

For the observations, we created a rota where one of the authors served as facilitator, and at least one other author participated as observer. After each observation session, both the facilitator and the observer filled in reflection notes in a template that contained the following

topics: group information; questions from the students on the technique; suggested changes to the game; participation; mood; topics discussed; what worked well with the game; challenges with the game, and; reflections on the observation and how the researchers may have influenced the process. After playing one session of Protection Poker, all groups were encouraged to keep on playing by themselves during the project, and we offered to return and offer support and/or facilitation at a later time, according to their needs.

When we facilitated the students in the capstone projects in playing the Protection Poker, we covered steps 1-6 in Figure 3 in addition to calibration. Each Protection Poker sessions lasted between 50 and 70 minutes and contained the following activities:

1. Introduction: The session started by having the students explain their system to the facilitator.
2. Assets: We prioritized calibrating the top end of the scale. The groups played on two to three assets, and spent between one and 17 minutes per asset played. For most (10 of 14) of the assets, the students were able to agree on a value with two rounds of playing the cards.
3. Features: Calibration of features was skipped in three of the groups due to limited time left. We prioritized playing about features over identifying and calibrating features. One group did not play on any features, because the nature of their project (creation of an algorithm) made it difficult to come up with features. The other five groups played on one to three features. The students spent between two and nine minutes per feature played. For all but one feature, two rounds of play were necessary.
4. Reflection: The session ended with reflection about the experience, and the students were asked to provide feedback and suggest improvements.

Throughout, the facilitator was active in helping the group reach a consensus by suggesting compromise values. This facilitation was done to speed up the playing, terminating discussions when most arguments had been raised.

Towards the end of the course, all groups were invited to send two to three participants to an event where the technique would be discussed in more detail. This event was organised as a group interview and was scheduled to last for two hours. The following topics were covered: students' expectations to the event; use of the game in the group; brainstorming and discussion on the 4Ls (Liked, Lacked, Learned, Longed for) (Caroli and Caetano, 2015); suggestions for improvements to the technique; suggestions for improvements to how software security was handled in the course, and; feedback on the event. Discussions were recorded and transcribed. To encourage participation, all participants were served pizza and they had the opportunity to win cinema gift cards. Non-responding groups were reminded via email. To promote active participation in the group interviews, each event was split in two parallel sessions. Note that in the observations we found that only two of the six groups had obvious security concerns. The group interview had low participation from those groups; only one participant from only one of those groups, while all the groups with limited security concerns participated with 2-3 people.

4.2 Industry practitioners

We have introduced Protection Poker to several of our industry collaborators, and in some cases we have been able to observe gameplay and collect questionnaires afterwards. This paper covers four interactions in companies and one interaction at a security conference aimed at industry. An overview of these interactions is given in Table 5. The questionnaires used for

data collection were a variation of the questionnaire used for the students, with the same TAM-based questions. The sessions (event 1-4) were organised in the following way:

1. Presentation of Protection Poker: The events all started with a presentation essentially identical to the one given to the capstone student groups.
2. Playing in groups, with facilitator: The groups played a few rounds, either on a synthetic case (event 2) or on features in their own backlog (event 1, 3 and 4). Note that in event 1 there were only two facilitators, so the other groups (one local and some remote) had to manage without support.
3. Reflections and questionnaire (in events 2 and 3). In event 1, the participants did not have time to fill out the questionnaire in the session but were asked to do this after the session and deliver to the security officer.

Table 5. Overview of events with industry practitioners

#	Where	Who	What	When	Data collected	Number of participants
1	Software development company	Developers, architects, security officer	Played PP on items in their backlog	June 2016 (2 hours)	Questionnaire responses (4), observation notes (2)	14 (3 groups), in addition to participants listening in from another site
2	Security conference for industry	Product Owner (1), manager (1), security architect (1), other security roles (10)	Played PP on synthetic case	September 2016	Questionnaire responses (13), notes from discussion with two facilitators	20
3	Software development company	Project manager and developers	Played PP on items in their backlog	October 2017	Questionnaire responses (5)	5
4	Software development company	Developers, security manager, architects	Played PP on items in their backlog	January 2017 (1,5 hours)	Observation notes (1)	8 on site, 1 listening in (one group)
5	Software development company	Security officer	Presented PP and got feedback on suitability for their organisation	June 2016	Notes (1)	1

4.3 Graduate-level software security course

Protection Poker was also studied in the graduate-level Software Security course (CSC515[1]) in the Computer Science department at the North Carolina State University during the autumn 2018 semester. This course is an optional elective for Master and PhD students. The purpose of the course is to introduce students to the discipline of designing, developing and testing secure and dependable software-based systems. During the fifteenth week of the semester, the students were given approximately a 30-minute lecture on the Protection Poker technique. After the lecture and during the same class period, the students did a 30-minute in-class exercise using the technique on a sample set of requirements.

In this course, the students are divided into 16 project teams (2-3 students per team, 58 students total). Through five deliverables during the 15-week semester, every team analyses the security of the OpenMRS[2] electronic health record application using a variety of techniques and tools. As part of the fourth deliverable, the students were asked to write five new functional requirements for Open MRS to add functionality that is not in the system yet. Each team then played Protection Poker on these requirements. This activity was not conducted in the presence of any of the teaching staff. After the deliverable was turned in, the students were asked to complete a questionnaire similar to that which was administered to the capstone groups. The students completed the questionnaire during class. The students were informed that the questionnaire response was part of a study and that their participation was optional, 46 students completed the survey. This study was approved by the university Institutional Review Board.

4.4 Analysis

All questionnaire data was analysed using descriptive statistics, to get an overview of responses to the individual questions. We did not do any further analysis of this data, since the number of responses from each part of the study was limited.

The qualitative data from the capstone development projects (observations, group interviews) was coded and organised within the Mind Manager tool. The qualitative data came in different forms, and the coding and organising of the data was first done for one type of data at the time. As an example, the template for observation notes contained a table to note what worked well and what was challenging with specific aspects of Protection Poker (Tøndel et al., 2018) and all the data collected there was analysed together, the same was the case for the results from the 4L exercise in the group interviews (Tøndel et al., 2018), to find what was common responses. Then the transcribed group interviews and the free text observation notes were coded together with the more structured qualitative data in the same mind map to extend the findings and to identify other topics. These activities resulted in the identification of main areas where improvement is needed (Tøndel et al., 2018), and the benefits and challenges of the Protection Poker taking only results from the capstone development projects into account.

The qualitative data from the interactions with industry representatives and from the study at the graduate security course was less extensive. The reflection notes from some of the industry events, and the responses given to the open-ended question in the questionnaire was coded in order to see if there were data there that supported or conflicted with the benefits and challenges identified in the capstone projects. In addition, we looked for challenges and benefits not already identified in the capstone projects.

[1] https://sites.google.com/a/ncsu.edu/csc515-software-security/
[2] https://openmrs.org/

5 Results

This section presents the results according to the two research questions of this study. Compared to Tøndel et al. (2018), this section gives an overview of results from industry events and a graduate course, in addition to the capstone projects. However, this section does not give as detailed an overview of the data collected from the capstone projects (observations, questionnaire results, 4L brainstorming and quotes from interviews) as the previous version. Readers who want more details on the results from the capstone project study are therefore referred to Tøndel et al. (2018).

The results point to one factor apart from Protection Poker that may have had a major impact on the results, namely the limited need for security in four out of the six capstone projects. The results relating to this factor is thus given special attention. An overview of the benefits and challenges identified with Protection Poker in this study can be found in Table 6 and Table 7.

Table 6. Overview of benefits identified, with an indication of where evidence is particularly strong, and where there may be limited evidence or no evidence

Benefit	Capstone projects	Industry	Graduate software security
Playing PP is perceived as useful (B1)	Learned things from playing PP. Helped them think about security. Gave overview of project (questionnaire, group interviews).	Expectation that PP can improve security (questionnaire).	***(Strong evidence.)*** *PP* is found to improve security (questionnaire).
PP is easy to learn (B2)	***(Strong evidence.)*** Most students agree that PP is easy to learn (questionnaire).	*(Limited evidence.)* Respondents are neutral on this aspect in the questionnaire responses.	Most agree that PP is easy to learn (questionnaire).
PP is easy to use (B3)	Most find PP easy to use (questionnaire). Observations and group interviews identify challenging aspects.	*(Limited evidence.)* Respondents are neutral on this aspect in the questionnaire responses.	Most agree that PP is easy to use (questionnaire).
PP brings about useful discussions in the team (B4)	***(Strong evidence.)*** View expressed in group interviews and supported by observation.	Supported by observation.	Discussions bring a deeper understanding and help reveal security issues (response to open-ended question).
PP makes everybody participate in security discussions (B5)	Supported by observations and group interviews. However, we observed that some are passive.	Observed that, although there was a homogenous group, they still ended up with different scores in the beginning. Observed in one company that team members did contribute to the discussion although they initially believed they did not know much.	Contributions from team-members with different perspectives are useful (response to open-ended question).
The relative scale makes PP useful also for projects where security is not a major issue (B6)	View expressed in group interviews	*(Not studied.)*	*(No evidence.)*

Results from playing PP are easy to interpret, and can be used for prioritization (B7)	View expressed in group interviews. Especially they liked the overview of the assets.	Liked the way PP estimate risk (response from one company representative).	PP can lead to risk reduction by modifying requirements (response to open-ended question)
PP is fun to play (B8)	View expressed in group interviews.	*(No evidence.)*	*(No evidence.)*
Increase knowledge and awareness about security (B9)	View expressed in group interviews.	Observed discussions that increased security knowledge among participants (example: A: "to attack would you not need to…" B: "but that is not so difficult because…")	*(No evidence.)*

Table 7. Overview of challenges identified, with an indication of where evidence is particularly strong, and where there may be limited evidence or no evidence

Challenge	Capstone projects	Industry	Graduate software security
PP poker did not improve security of the software (C1)	***(Strong evidence.)*** Results from PP does not directly influence development (questionnaire, group interviews).	*(Not studied)*	***(Conflicting evidence.)*** Improved security is considered a main benefit from PP (questionnaire) and students express that playing PP has made an impact through modifying requirements (response to open-ended question)
Limited relevance for their project (C2)	Four of six projects had few security concerns.	*(Not studied.)*	*(Not studied.)* The OpenMRS had many security concerns.
Starting to use PP is time consuming due to calibration and the need to identify and play about assets (C3)	***(Strong evidence.)*** Supported by observations. The time needed to play was considered a challenge in group interviews.	Observed that calibration and asset identification for the first feature took a lot of time	*(No evidence)*
It is difficult to reach consensus, something that results in a lot of time spent and sometimes result in tension in the team (C4)	View expressed in group interviews and supported by observations. But in some of the group there were no problems related to this (observations).	*(No evidence.)*	One student expressed challenges related to conflicts (response to open-ended question)
Some team members may end up with too much influence (C5)	View expressed in group interviews and supported by observations.	*(No evidence.)*	*(No evidence.)*
Ensuring confidence in the result (C6)	View expressed in group interviews	*(No evidence.)*	*(No evidence.)*

The relative scale can be difficult to understand (C7)	Observed in one group. View expressed in group interview.	*(No evidence.)*	*(No evidence.)*
Selecting granularity of assets and assigning value to assets can be challenging (C8)	View expressed in group interview.	Some expressed that assets should be identified beforehand to ensure they are at a common level.	*(No evidence.)*
The term exposure is difficult to understand (C9)	Observed in several groups.	Observed in one of the companies.	*"It's hard to know what is easy to access and what is not"* (response to open-ended question).
Exposure and asset value are often mixed up in the discussions (C10)	***(Strong evidence.)*** Many questions on the terms, and the terms were often mixed up in discussions (observations).	Observed in at least two events. Possible misunderstanding: exposure for each asset.	*(No evidence.)*
Important aspects from the discussion is lost (C11)	Observed in several groups.	In one company it was suggested to improve this by having something on Jira.	*(No evidence.)*
Teams did not end up using PP in a regular fashion (C12)	***(Strong evidence.)*** Only one of the student groups used PP on their own, and then only once.	*(Not studied.)*	*(Not studied.)*
Planning meetings are already full (C13)	*(Not studied)*	Response from discussing Protection Poker with company representatives	*(Not studied.)*
Scalability across teams (C14)	*(Not studied)*	In one company it was pointed out that features and tasks could be moved across teams, and that they had a common backlog. Then it would be easier if all teams used the same assets and had the same asset values for them.	*(Not studied.)*
Many cards (C15)	Response from some students in group interviews.	Response from some players in one of the companies.	*(No evidence.)*
The output from playing PP is not concrete in terms of what to do next (C16)	Lacked discussions on how an attack could happen (group interviews).	In the observation notes from one company it was noted the lack of a "practical product" from the meeting.	*(No evidence.)*
PP takes too much time (C17)	Observed when beginning to use PP, but do not know how this will be later. Questionnaire responses slightly disagree. See also C4.	Neutral questionnaire responses. But concerns that teams will not be able to use the technique (ref. C13)	One student stated that PP was *"cumbersome and time consuming"*, another that it was more formal than what would be expected in companies (response to open-ended question). Neutral questionnaire responses.

5.1 Acceptance of Protection Poker (RQ1)

Acceptance of Protection Poker (RQ1) was mainly studied through the TAM-based questionnaire. Table 8 gives a high-level overview of the responses to the questionnaire. Detailed results can be found in Tøndel (Tøndel, 2018). The responses come from different sources:

- Responses from the students of the capstone project after the introductory lecture (29 responses)
- Responses from the students in the capstone projects at the end of the course (30 responses)
- Responses from industry representatives taking part in playing Protection Poker at different events (21 responses)
- Responses from students in a graduate-level course after using Protection Poker to analyse the security risk of new requirements for their semester-long project (46 responses)

Table 8. High level overview of responses to questionnaires

TAM variable	Capstone projects – *before*	Capstone projects – *after*	Industry representatives	Graduate software security
Future use intention	Somewhat agree	Somewhat agree, but less so	Somewhat agree	Agree
Perceived usefulness	Somewhat agree	Neutral	Somewhat agree	Agree
Perceived ease of use	Somewhat agree	Agree	Neutral	Somewhat agree

Four questions together cover the variable *future use intention* (I intend to increase my use of the PP for project-work in the future; I intend to use the PP in the future for my projects; Given a choice, I would prefer not to use the PP in any future projects; I would like to use the PP in the future). In the capstone projects responses show that the students tend towards being positive to use Protection Poker. In the end, half (15) of the capstone project students agree that they would like to use Protection Poker in the future, while only five did not want to use Protection Poker. Among the industry representatives the responses are quite similar, with twelve wanting to use Protection Poker, while only two did not want to use Protection Poker. The graduate students in the software security class were the most positive to using Protection Poker, where 38 agree that they want to use Protection Poker, while only 3 disagree.

Four questions together cover the variable *perceived usefulness* (I think PP will be useful in my current project; Using the PP will improve the security of the product; Using the PP will substantially reduce the number of serious security defects; The advantages of using the PP outweighs the disadvantages). Many students in the capstone projects found Protection Poker to be useful; in the end more students agree (14) than disagree (4) that the advantages of using Protection Poker outweigh the disadvantages. Expectations among the capstone project students on what Protection Poker would deliver was in general high, however, it seems that Protection Poker did not quite deliver in their current project. In particular, Protection Poker does not seem to have delivered on security; not improving the security of the product and not reducing security defects. The responses from the industry representatives are very much in line with the responses from the students before they had used Protection Poker in their own project, with one exception: the industry representatives are more positive regarding the results

from playing Protection Poker on the security of the product. On the question *Using the PP will improve the security of the product*, twelve of the 21 respondents agree, and four strongly agree, while only one disagrees. The graduate students in the software security class are even more positive on the usefulness of Protection Poker, and especially on its ability to improve security; 25 students agree, 12 strongly agree and none disagree.

Six questions together cover the variable *perceived ease of use* (Learning to use the PP was easy for me; I think the PP is clear and understandable; Using the PP requires a lot of mental effort; I find the PP easy to use; The PP is cumbersome to use; Using the PP takes too much time from my normal duties). Overall, the capstone project students' responses to these questions are positive, and increasingly so towards the end of the course. To illustrate, in the end only one of the capstone project students found Protection Poker to be difficult to learn, as opposed to 25 finding it easy, and the majority of the students ended up finding Protection Poker to be clear and understandable (22 students) and easy to use (23 students). The responses from the industry representatives however show that they did not find Protection Poker to be as easy to use, and their responses were overall neutral on the corresponding questions. The graduate students in the software security class responses generally lie between these two groups.

5.2 Lessons learned and improvements identified by the players (RQ2)

Lessons learned and improvements (RQ2) were studied through observations and group interviews in the study of the capstone projects. Two main areas were identified where improvements are needed; the discussions, and the scores and scales used. In the following, we introduce these areas and the improvements suggested by the students related to these areas.

The discussions resulting from playing Protection Poker was considered highly useful by many of the students that participated in the group interviews, but at the same time they reported on several challenges related to keeping the discussions effective and efficient. Key concerns by the students was that: 1) some players end up with too much influence due to their personality; 2) difficulties in reaching consensus results in fighting instead of a common understanding; and 3) Protection Poker sessions take time. These benefits and challenges were supported by observation notes from the researchers as well. In the observation notes, half the capstone groups (3) were characterized either by dominant of passive participants, something that negatively influenced the general mood while playing. In all groups, the facilitator was quite active in supporting the students in reaching a consensus. An additional challenge observed was that important aspects from the discussions got lost as it was not noted down anywhere.

Although students did not have any clear suggestions for improvements that directly address the challenge of having both good discussions and efficient playing of Protection Poker, they had suggestions that can partly help improve the challenges related to the discussions. One suggestion was to have fewer cards, and thus a more coarse-grained scale. Another suggestion was to have more support on security in form of what to discuss and how to ensure they were on the right track. Both the response from the students after the sessions and our own observations suggest that it would have been very difficult for the students to start using Protection Poker without an external facilitator that could help on the game and bring in software security competence. Though the need for an external facilitator was clearly expressed by the students, it is important to add that one group played Protection Poker on their own after the supported session, and they reported that this had gone very well, and in some ways better since they did not have to explain the system to someone external.

Challenges relating to scales and scores concerned two main issues: understanding the relative scale, and understanding the concepts *asset value* and *exposure*. Having a relative scale was considered a benefit by some students, as it made Protection Poker a useful technique also for projects without any major security issues. However, the scale was considered difficult to understand and relate to by other students; *"You did not know if a '100' was Armageddon or it was just "we need to look into this"."* Additionally, disagreements on how to understand the scale slowed down playing in some groups. Some of these challenges that we experienced with the scale may be related to us skipping calibration of the low end of the scale in order to save time. The improvements suggested by the students related to the scale go in two directions: to explain the relativity of the scale better, or to change the scale. The latter suggestion was less common. Additionally, students suggested to take the time to do a full calibration.

The terms *asset* and *exposure* seemed to be new to many of the students, and in the observations the students had many questions on these terms. The terms were often mixed up in the discussions, with students talking about the exposure of an asset or the value of a feature. Especially the term exposure was found difficult to describe in a good way. For assets it was sometimes difficult to know how to assess their value, as the value may be different if you consider just confidentiality than if you include other aspects of its value as well. Another challenge identified by the students was how to divide up assets in a way that is consistent and does not impact the scores in an unintended way. They pointed out that if you have assets at different levels of granularity this may skew the scores; a feature with many assets of low granularity may get a higher score, and thus priority, than a feature that has assets with higher granularity. Note however that despite these challenges, students expressed that they found the end result to be easy to interpret, that is was predictable due to the process and that it gave them a nice way to prioritize the assets of the project. Students did not suggest any changes to these terms, but they suggested to identify assets in advance, and provide better guidance on how to identify assets.

5.3 Effect of limited security issues in the capstone projects

The capstone project study found that students perceived little benefits from playing when it comes to security. The open-ended responses on the questionnaire shed some light on this. Though students did expect Protection Poker to have benefits, they were divided in their expectations. Ten out of the 28 that responded to the open-ended question "*How do you think playing PP will influence the product?"* stated that they did not expect much influence. Of those that did expect an influence, the majority (11) expected it to improve security awareness. Other expectations included identifying the most important parts regarding security (4), a more secure product (3), discussions on security (2) and agreement on security issues in the group (2). Those that explained why they did not expect an impact from playing the game, explained that this was due to limited security issues in their project. Open ended responses to the question *"How do you believe software security is important to your project?"* confirm our observations that only two of the six groups had clear security issues that needed to be dealt with, while four groups had very limited attack surfaces or assets of little value, thus having limited security needs. Since it is likely that the limited security needs of projects influenced the usefulness of the technique when it comes to security, we additionally looked at the responses from the two groups that had security issues (10 responses) in isolation. We found that for the question on whether Protection Poker will improve the security of the product, all five students that agree that using Protection Poker improved security come from these two groups. When it comes to reduction of security defects, the students from the groups with security issues are more positive than the others, however, also these students in general do not agree that they experienced such a reduction from playing Protection Poker. Note that the graduate students in

the software security are more positive regarding the effect of playing Protection Poker on security. These students both had more security competence and a project where security was an important part because patient medical records are involved.

Despite limited need for security, the questionnaire responses indicate that students still ended up being quite positive regarding the usefulness of the game. Positive aspects of the game were discussed in the group interviews, and these can shed some light on what the students found useful. Overall, the students were positive to security and see the need for it in the general case. They explained that they learned many things from playing. This included knowledge about security (assets, attack surface, easy to overlook security issues). However, other more general insights were more often pointed out, such as gaining experience in group discussions, making decisions, coming to consensus, etc., and that they learned things about their own software projects and how it was understood by other group members. As stated by one student in the group interviews, *"I think it is a good game, I think it works fine, but I don't think I got that much out of it as I could have, and I could have learned more about the different parts of Protection Poker and software security if I had a game or a project with more security issues."*

6 Discussion

In our interaction with industry and in the student projects we find that people generally react positively to Protection Poker when we introduce the game but there is room for improvements. As can be seen from Table 6 and Table 7, industry representatives and students that used Protection Poker in different types of projects ended up experiencing many of the same benefits and challenges with Protection Poker. The main differences are as follows:

- Industry representatives and students doing a security course are more positive on the effects Protection Poker has on security
- Industry representatives find Protection Poker more difficult than the students (to learn and use)
- Less challenges related to the discussion was observed with industry representatives
- Industry representatives identified challenges that was not relevant for the student projects, e.g. challenges with scalability and with integrating the playing of Planning Poker with other planning activities in industry settings.

Many of the challenges identified with Protection Poker in this study overlap with challenges identified in literature when it comes to integrating security activities in agile development (see Section 2.2); this includes challenges on

- running the meetings (Cruzes et al., 2018): e.g. challenging discussions and group dynamics (C4-5 in Table 7)
- documentation (Cruzes et al., 2018): e.g. that the end result is not concrete enough (C16) and that key aspects from the discussion is lost (C11)
- integrating security techniques into the development work (Türpe and Poller, 2017): e.g. gaining impact from playing (C1) and integrate with other activities and way of work (C13, C14)
- priorities (Terpstra et al., 2017): e.g. making the time needed acceptable (C3, C17)
- awareness and knowledge (Terpstra et al., 2017): e.g. making security terms understandable for the players (C9-10) and ensuring confidence in the results (C6)

Due to the overlap in types of challenges found, many of the issues that end up being challenging with Protection Poker may not be due to this particular technique, but rather point

to more general challenges with this type of work. Thus, finding ways to tackle such challenges with Protection Poker can be useful input also to other techniques in this domain.

Based on the findings from this study, we would point at four major issues that need to be improved on Protection Poker:

- Making the time needed to play Protection Poker more acceptable for the teams;
- Ensuring impact from playing Protection Poker on the security of the end-product;
- Better integrate Protection Poker with project planning activities;
- Clarify the scenarios for better adoption of Protection Poker in a project.

The students in the capstone projects ended up finding that Protection Poker did not improve the security of the product (C1). Even though the professionals that participated in this study and the graduate students were more positive on this aspect of Protection Poker, such a feedback on the technique needs to be addressed, as a goal of improved security is the main reason for investing time in performing such a technique in the team. Additionally, we got the feedback both from students and professionals that playing Protection Poker took too much time (C3-4, C17), and professionals pointed out the difficulties in integrating such a time-consuming activity into existing planning activities in the projects (C13). As time and budget is considered important factors that influence adoption of software security practices (Kanniah and Mahrin, 2016) (see Section 2.2), it is important to address challenges related to the time needed. Results additionally point to Protection Poker being more useful in some of the teams than in others, thus teams considering adopting Protection Poker should be aware of the factors that can impact the usefulness of the technique for their particular situation. In the following we discuss how to improve these parts of the game. An overview of the approach we are suggesting can be found in Figure 3.

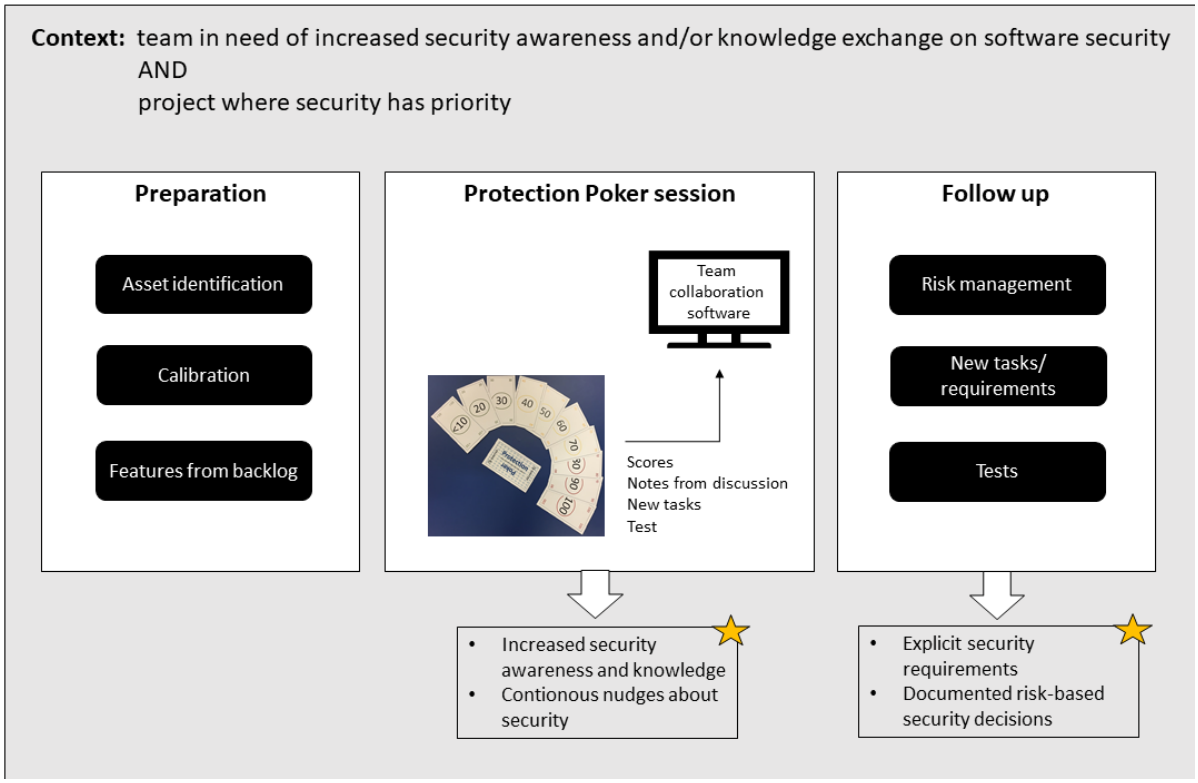

Figure 3. Overview of our suggested approach to adopting Protection Poker

6.1 Reduce time needed to start using Protection Poker

Other studies have shown that the time needed to play Protection Poker is reduced after it has been used by the team for some time (Williams et al., 2010). Still, it is important to ensure that teams considering to adopt Protection Poker are not put off by the time it takes to start using Protection Poker the first time (C3). Based on our experiences in these studies, the long time it takes to begin using Protection Poker is due to the following issues:

- The time needed to learn the technique itself
- The time needed to become familiar with the terms 'asset', 'asset value' and 'exposure'/'ease of attack'
- The need to calibrate the scale
- The need to play about all asset values related to a feature

There is nothing that points to Protection Poker being more time consuming to learn than other techniques, in fact one benefit identified is that Protection Poker is easy to learn (B2 in Table 6). In the study, some challenges on understanding the terms were identified (C9-10). However, this would probably be the case also for other security techniques if the players are unfamiliar with software security (as was the case for most of the students in the capstone projects). Though Protection Poker may benefit from terms that are even more easy to use and explain, the terms 'asset' and 'exposure' (or variations thereof) are common in security techniques. Thus, it is likely that the main improvements related to Protection Poker and the time it starts to use the technique would be related to calibration and playing about asset value.

In the capstone projects, we skipped part of the calibration in order to save time. Although we observed that this worked well in most of the groups, we got the feedback from the students that they would recommend taking the time to do a full calibration in future projects. Identifying and prioritizing assets was something that the capstone project students in general found useful (B7), however, there were challenges associated with doing this as part of playing Protection Poker: it took time and there was the concern that if this was not done at a similar granularity for the whole project this might skew the prioritization one ended up making through the game (C8).

There are various options to reduce the time needed to do calibration and asset identification. Teams could:

- Decouple asset identification from the playing of Protection Poker.
- Let one or two people perform calibration as a preparation.
- Drop the division into assets and exposure/ease of attack altogether, and play about only one issue per feature, e.g. 'How easy is it to misuse this feature to get to/harm important assets?'

Decoupling of asset identification from the playing of Protection Poker was suggested by both students in capstone projects and by professional developers in one company. As this is likely to speed up playing, as well as limit the challenge related to granularity of assets (C8), we suggest that teams take this approach. Regarding calibration, we expect that this could be done as a preparation, as we did not have major challenges in most teams where full calibration was dropped. This would allow teams to spend less time on calibration, but still have a scale that is calibrated for their specific project (B6). Regarding the alternative of dropping the division into assets and exposure altogether, we have no data to indicate whether or not this would be an improvement to the game. Thus, we do not recommend that teams take this approach. However, we would welcome more research into what are the most important questions to ask in a

security analysis task. We observed that players mixed up the terms assets and exposure and had challenges in understanding these terms (C9-10). Thus, it may be just as easy for them to drop this division and consider them together. However, one important thing that the capstone project students liked was to get an overview of the assets for the project (B7), and if going in this direction one would lose that benefit of playing Protection Poker.

6.2 Gaining impact from playing Protection Poker

Apart from limited security needs in four of the capstone projects (C2), we do not know what factors that potentially made it difficult to use the results from the Protection Poker game in the development. In the capstone projects, it was up to the students to use the results from playing in any way they found fit. We did not follow up on how they used the results, and provided no specific guidance on how to do this. One potential issue is the limitation pointed out by one student in the group interview that the game does not include anything on *how* such an attack can be mitigated. If students lack this knowledge it can be difficult to understand what can be done to reduce the risk associated with what they consider high risk functionality in the software. The results from playing Protection Poker is a prioritization, something the students found to be an important benefit (B7), however turning this prioritization into actual development tasks is not necessarily straight forward, especially if the rationale for the scores is lost due to limited note taking (C11).

A major benefit of the Protection Poker technique is that it involves all team members in useful discussions (B4-5). However, for these discussions to have an impact on the end-product, it is important that key aspects from these discussions are documented (C11) and made into actionable tasks (C16). To encourage the documentation of more aspects from the discussions, we recommend that teams have team collaboration software open during the playing of Protection Poker, and that one person (e.g. the facilitator, the security champion of the team, or someone else appointed this task) is responsible for adding important issues along the way. This includes open issues that need to be investigated further, suggestions for mitigations, attack vectors that should be considered, new assets identified, opinions on the whether the risk is acceptable or not, ideas for tests, etc. This way one ensures that the team is not only left with a score after the discussions, but also have the rationale behind the score. After the session where Protection Poker is played, it is essential that someone is responsible for making decisions based on the result from playing Protection Poker, and enforcing these decisions by adding development tasks.

6.3 Integrate Protection Poker with project planning activities

A general impression, after observing the playing of Protection Poker and talking with practitioners is that few teams would be willing to adopt Protection Poker for every iteration. Many of the main benefits from the technique as identified in this study, such as good discussions (B4), increased security awareness (B9), overview of system (B1), etc. is not dependent on playing Protection Poker for every feature. However, if not playing about every feature then teams need other ways to ensure that security is addressed in a holistic manner, and teams may lose the benefit of having regular security discussions in the full team. One option to solve this challenge is to play Protection Poker less regularly, and for more high-level features. Instead of playing Protection Poker for every feature that is to be implemented in an iteration, one could play Protection Poker for the main features in each epic (more overall groups of functionalities). Playing Protection Poker for each epic comes with the benefit that one can more easily use the result of playing Protection Poker as input to effort estimation and prioritization. Alternatively, teams would need some criteria as to when a round of Protection

Poker should be played. Teams could also decide to use Protection Poker entirely as an awareness and training tool at various points throughout the project, discussing the features that are more relevant for security in that stage of the project. Then other techniques are necessary to ensure the elicitation of security requirements.

6.4 Criteria for adopting Protection Poker in a project

The results from this study show that Protection Poker is not that useful for projects with very limited security issues, either because of very limited attack surface or few assets of any particular value. This type of projects is probably not as prominent in development companies as in our case with capstone projects. Still, our results point to the need for some kind of criteria to evaluate whether there are enough security issues in a project to justify the effort needed to play Protection Poker.

The discussions were a main source of the benefits from playing Protection Poker, but also a source of challenges, especially concerning time. Due to team dynamics issues, the teams experienced the playing of Protection Poker quite differently. Protection Poker initially aims to support good discussions, and the voting involved when putting out a card is a way to ensure that all team members' opinions are made visible. However, the goal of reaching a consensus is not realistic in many settings. Teams need to be made aware that this is challenging, and not necessarily a strict goal. Though it is not beneficial to have everybody always agree, playing Protection Poker with participants that *never* agree, or always need to be right, is challenging. Based on the results from this study one can assume that how well Protection Poker will perform in the team, and especially the efficiency of the discussions, is highly influenced by the team dynamics. Knowledge of team dynamics could thus be one factor to consider when deciding if Protection Poker would be a good technique for a particular team.

One reason why Protection Poker takes time is that it is recommended that the full team participates in the playing. Having only one or two persons make an evaluation of assets related to a feature, their value, and the exposure related to a feature, would be more efficient in terms of time. However, it would be hard to get the same awareness raising in the whole team related to security (B9) (Williams et al., 2010) with such an approach. Thus, teams that already have a high awareness and knowledge about software security may not need to spend the extra time on playing Protection Poker in the full team. However, if awareness and knowledge raising is needed on security, playing Protection Poker could be considered a type of security training for the whole team, and thus the extra time may be well worth the effort in the long run. Organisations and teams should consider this before deciding whether to invest time in playing Protection Poker.

7 Threats to validity

This study involves both students and professional software developers, however, most of the data comes from student projects. By performing the major parts of the study in a university setting we were able to control the setup of the study in a way that would be difficult to do with companies. Additionally, we had the ability to collect more data, since the time students needed to invest in the data collection activities were less of a concern than what would be the case for professional developers. This allowed us to use several data collection methods to increase confidence in the results. Still, performing the study with students has its drawbacks as these have different experiences and are in a different context than professional software developers.

Research has shown that students in the later parts of their studies can be used with success in studies instead of professional software developers in some cases, namely for understanding dependencies and relationships in software engineering [11] and for requirements selection [24]. The topic of this study is related to, but not identical to, those studies. We do not claim that the results from our study can be generalised to software developers in general, but believe it to be likely that many of the same issues that we found would apply also in professional settings, in particular since many professionals in small and medium sized development organisations would also be considered novices when it comes to Protection Poker and have limited software security training [12]. However, the context would be different. Although the students in the capstone projects did have an external customer and the aim of the course is to have a setting that is as similar as possible to a real development project, the students had some concerns that professionals would not have (e.g. writing the report and getting a good grade) and this may have impacted the results. Their development projects were also likely to be simpler and with fewer security concerns than what many professional developers would likely encounter. From the results of our study we can see that many of the same benefits and challenges were observed with both students and professionals, but it is important to note that there were main differences as well. Thus, a study with more participation from professionals may have yielded different results on some aspects.

In the capstone project study, it is difficult to separate the effect of the technique itself from other factors, such as motivation, skills, group dynamics, and our influence as researchers. In particular, having researchers act as facilitators constitutes a threat to validity that may influence the process and the results and make the study harder to replicate. We have aimed to be aware of the impact of the context throughout the capstone project study. One way we did this is by having the first author be supervisor of one student group. Additionally, we made sure we reflected on our role as researchers and took this into account in the analysis (reflection on our influence as researchers was part of the template for observation notes). As part of this, we made it clear for students that their opinion on Protection Poker would not have any impact on their grade in the course. We as researchers did not have any influence on the grades the students got, except for giving some input to evaluators for the group where the first author acted as supervisor.

In the events with professionals, our ability to collect rich data was reduced, and these events were more limited in time than what was the case with students. The companies were recruited based on existing contacts. Additionally, the response rates on the questionnaires were sometimes rather low. Both these factors may lead to more positive responses regarding Protection Poker than what we may have gotten in other companies and with more responses, as it is likely that the more positive companies and participants when it comes to security are willing to participate. The participants in the session in the conference mainly had roles and background relating to security, and limited development background. Security people have a potentially important role in promoting and supporting security work in software projects (Tøndel et al., 2017), thus the opinions of representatives with this role is relevant to consider. However, their opinions and perspectives may differ from developers. We did not aim to analyse differences between responses from different groups of professionals, since the number of questionnaire responses was not large enough to justify such analysis.

In the graduate level course, students performed the Protection Poker activity on their own and reported the results. They were not aware they would be completing a questionnaire when they completed the assignment, and the student were made aware questionnaire was anonymous. When asked about their participation on the Protection Poker part of the assignment,

approximately 10% of the students indicate they did not participate fully yet they answered the questionnaire. There were three other parts of the assignment, and sometimes the students divided up the work throughout the group.

8 Conclusion

Protection Poker is a collaborative technique for risk estimation that is particularly suited for agile development teams. As of now we are not aware that Protection Poker ends up being adopted by teams. This study has identified both benefits and challenges with Protection Poker. It suggests how to tackle the main obstacles to adoption of the technique, including ways to address the challenge that teams may find that Playing Protection poker takes more time than they are willing to spend, at least in every iteration. Additionally, it points to the importance of finding ways to ensure playing Protection Poker ends up having an impact in form of improved security of the end-product. To tackle this, this paper suggests increasing preparation activities, and ensuring more documentation and follow-up from the discussions that take place during the playing.

Acknowledgments

This work was supported by the SoS-Agile: Science of Security in Agile Software Development project, funded by the Research Council of Norway (grant number 247678). Thanks to the course organisers of TDT4290 (Prof. Jon Atle Gulla and Prof. John Krogstie) and the participating students at NTNU and North Carolina State University. Thanks to Tosin Daniel Oyetoyan for contribution to the capstone study. Thanks to Prof. Pekka Abrahamsson for input on the capstone study design. Thanks also to the companies that participated in the events, and to those helping with facilitation at the security conference (Per Håkon Meland and Marie Moe).

References

AJZEN, I. & FISHBEIN, M. 1980. *Understanding attitudes and predicting social behaviour*, Pearson.

ALSAQAF, W., DANEVA, M. & WIERINGA, R. 2017. Quality requirements in large-scale distributed agile projects–a systematic literature review. *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer.

BACA, D., BOLDT, M., CARLSSON, B. & JACOBSSON, A. 2015a. A novel security-enhanced agile software development process applied in an industrial setting. *Availability, Reliability and Security (ARES), 2015 10th International Conference on*. IEEE.

BACA, D., BOLDT, M., CARLSSON, B. & JACOBSSON, A. A Novel Security-Enhanced Agile Software Development Process Applied in an Industrial Setting. 2015 10th International Conference on Availability, Reliability and Security, 24-27 Aug. 2015 2015b. 11-19.

BECK, K., BEEDLE, M., VAN BENNEKUM, A., COCKBURN, A., CUNNINGHAM, W., FOWLER, M., GRENNING, J., HIGHSMITH, J., HUNT, A. & JEFFRIES, R. 2001. Manifesto for agile software development.

CARALLI, R. A., STEVENS, J. F., YOUNG, L. R. & WILSON, W. R. 2007. OCTAVE Allegro: Improving the Information Security Risk Assessment Process (CMU/SEI-

2007-TR-012 ESC-TR-2007-012). *Software Engineering Institute at Carnegie Mellon University*.
CAROLI, P. & CAETANO, T. J. L., LAYTON 2015. Fun Retrospectives-Activities and ideas for making agile retrospectives more engaging.
CHANDRA, P. 2008. Software assurance maturity model. *A guide to building security into software development v1. 0, OWASP Project.*
CRUZES, D. S., JAATUN, M. G., BERNSMED, K. & TØNDEL, I. A. 2018. Challenges and Experiences with Applying Microsoft Threat Modeling in Agile Development Projects. *Australasian Software Engineering Conference (ASWEC).* Adelaide, Australia.
CYBENKO, G. 2006. Why Johnny Can't Evaluate Security Risk. *IEEE Security & Privacy***,** 5.
DAVIS, F. D. 1985. *A technology acceptance model for empirically testing new end-user information systems: Theory and results.* Massachusetts Institute of Technology.
DAVIS, F. D. 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly***,** 319-340.
DELEERSNYDER, S., WIN, B. D. & GLAS, B. 2017. Software Assurance Maturity Model - How To Guide - A Guide to Building Security Into Software Development. 1.5 ed.: OWASP.
DINGSØYR, T., MOE, N. B., FÆGRI, T. E. & SEIM, E. A. 2018. Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. *Empirical Software Engineering,* 23**,** 490-520.
FENZ, S. & EKELHART, A. 2010. Verification, validation, and evaluation in information security risk management. *IEEE Security & Privacy***,** 58-65.
GEER, D. 2010. Are companies actually using secure development life cycles? *Computer,* 43**,** 12-16.
GERBER, M. & VON SOLMS, R. 2005. Management of risk in the information age. *Computers & Security,* 24**,** 16-30.
GRENNING, J. 2002. Planning poker or how to avoid analysis paralysis while release planning. *Hawthorn Woods: Renaissance Software Consulting,* 3**,** 22-23.
HOWARD, M. & LIPNER, S. 2006. *The Security Development Lifecycle:: A Process for Developing Demonstrably More Secure Software*, Microsoft Press.
ISO/IEC 2011. ISO/IEC 27005: 2011Information technology–Security techniques–Information security risk management.
JAATUN, M. G. & TØNDEL, I. A. 2008. Covering your assets in software engineering. *The Third International Conference on Availability, Reliability and Security.* Barcelona: IEEE.
JAATUN, M. G. & TØNDEL, I. A. 2016. Playing Protection Poker for Practical Software Security. *International Conference on Product-Focused Software Process Improvement.* Springer.
JOURDAN, Z., RAINER, R. K., MARSHALL, T. E. & FORD, F. N. 2010. An Investigation Of Organizational Information Security Risk Analysis. *Journal of Service Science,* 3**,** 10.
KANNIAH, S. L. & MAHRIN, M. N. 2016. A Review on Factors Influencing Implementation of Secure Software Development Practices. *International Journal of Computer and Systems Engineering* [Online], 10. Available: http://doi.org/10.5281/zenodo.1127256.
KHAIM, R., NAZ, S., ABBAS, F., IQBAL, N. & HAMAYUN, M. 2016. A review of security integration technique in agile software development. *International Journal of Software Engineering Applications,* 7.

LI, L. 2010. A critical review of technology acceptance literature. *Southwest Decision Sciences Institute Conference.*
MCGRAW, G. 2004. Software security. *IEEE Security & Privacy,* 2**,** 80-83.
MCGRAW, G. 2006. *Software security: building security in*, Addison-Wesley Professional.
MCGRAW, G., MIGUES, S. & WEST, J. 2016. Building Security In Maturity Model (BSIMM7). Cigital.
MICROSOFT. 2012. *Security Development Lifecycle for Agile Development* [Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/ee790621.aspx [Accessed].
NIST 2010. Guide for Applying the Risk Management Framework to Federal Information Systems - A Security Life Cycle Approach. R1 ed.
NYFJORD, J. & KAJKO-MATTSSON, M. 2008. Integrating Risk Management with Software Development: State of Practice *International MultiConference of Engineers and Computer Scientists* Hong Kong: IAENG.
ODZALY, E. E., GREER, D. & STEWART, D. 2017. Agile risk management using software agents. *Journal of Ambient Intelligence and Humanized Computing*.
OUESLATI, H., RAHMAN, M. M. & BEN OTHMANE, L. 2015. Literature review of the challenges of developing secure software using the agile approach. *2015 10th International Conference on Availability, Reliability and Security (ARES).* IEEE.
RHEE, H.-S., RYU, Y. U. & KIM, C.-T. 2012. Unrealistic optimism on information security management. *Computers & Security,* 31**,** 221-232.
SULAMAN, S. M., WEYNS, K. & HÖST, M. A review of research on risk analysis methods for IT systems. Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering, 2013. ACM, 86-96.
TAVARES, B. G., DA SILVA, C. E. S. & DE SOUZA, A. D. 2017. Risk management analysis in Scrum software projects. *International Transactions in Operational Research***,** n/a-n/a.
TERPSTRA, E., DANEVA, M. & WANG, C. 2017. Agile Practitioners' Understanding of Security Requirements: Insights from a Grounded Theory Analysis. *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW).* IEEE.
TÜRPE, S. & POLLER, A. 2017. Managing Security Work in Scrum: Tensions and Challenges. *International Workshop on Secure Software Engineering in DevOps and Agile Development.* Oslo, Norway: CEUR-WS.
TØNDEL, I. A. 2018. Results from questionnaires on Protection Poker.
TØNDEL, I. A., JAATUN, M. G., CRUZES, D. & OYETOYAN, T. D. 2018. Understanding challenges to adoption of the Protection Poker software security game. *2nd International Workshop on SECurity and Privacy Requirements Engineering (SECPRE 2018).* Barcelona.
TØNDEL, I. A., JAATUN, M. G., CRUZES, D. S. & MOE, N. B. 2017. Risk Centric Activities in Secure Software Development in Public Organisations. *International Journal of Secure Software Engineering,* 8**,** 1-30.
TØNDEL, I. A., LINE, M. B. & JOHANSEN, G. 2015. Assessing information security risks of AMI: What makes it so difficult? *1st International Conference on Information Systems Security and Privacy 2015.* Angers, France.
VENKATESH, V. & DAVIS, F. D. 1996. A model of the antecedents of perceived ease of use: Development and test. *Decision sciences,* 27**,** 451-481.
WILLIAMS, L., GEGICK, M. & MENEELY, A. 2009. Protection poker: Structuring software security risk assessment and knowledge transfer. *International Symposium on Engineering Secure Software and Systems.* Springer.

WILLIAMS, L., MCGRAW, G. & MIGUES, S. 2018. Engineering Security Vulnerability Prevention, Detection, and Response. *IEEE Software,* 35**,** 76-80.
WILLIAMS, L., MENEELY, A. & SHIPLEY, G. 2010. Protection Poker: The New Software Security "Game". *IEEE Security and Privacy,* 8**,** 14-20.