Heidi Tonteri

“The law that mandates us is stronger than the consumer's rights”: what are the decisions related to authentication method selection?

Master’s thesis of mathematical information technology

May 15, 2023

University of Jyväskylä

Faculty of Information Technology

Author: Heidi Tonteri

Contact information: heidi.e.tonteri@gmail.com

Supervisors: Hanna Paananen and Naomi Woods

Title: “The law that mandates us is stronger than the consumer's rights”: what are the decisions related to authentication method selection?

Työn nimi: “Meitä velvoittava laki on vahvempi kuin kuluttajan oikeus”: minkälaista päätöksentekoa tunnistautumismenetelmän valintaan liittyy?

Project: Master’s thesis

Study line: Software and Telecommunications Technology

Page count: 63+3

Abstract: There exist various frameworks for the selection of end-user authentication methods, but none of those takes a stand concerning the organizational point of view regarding secure software development. The purpose of this research was to gain insight into secure software development and explain how security features are implemented in the developed systems. The research was carried out using a qualitative method, conducting semi-structural interviews for seven participants from Finnish IT organizations. Data were processed by thematic analysis. The theoretical framework was Anthony’s (1964) organizational decision-making processes, and it was used in analysing the data. The research shows that different security features are implemented at different phases of secure software development. Security policies created at the company’s strategic level are transferred to technical instructions that guide architectural decisions, requirements engineering, the selection of authentication methods, and component integration. Contradicting Anthony’s theory, developers’ influence on decision-making is notable; developers as experts can oversee high-level technical decisions such as relating to the architectural resolution. They may also use power to gain changes counteracting official company policies such as the selection of development methods. The study shows that high regulation is the main source of requirements for companies and the selection criteria of authentication method is similar than relating to any component.

It also shows that system developers do not take responsibility for maintaining the current authentication practice due to the reliance on regulation.

Keywords: Secure software development, cyber security, security features, end-user authentication

Suomenkielinen tiivistelmä: Loppukäyttäjän tunnistusmenetelmien valinnalle on olemassa erilaisia viitekehyksiä, mutta yksikään niistä ei ota kantaa organisaation näkökulmaan turvalliseen ohjelmistokehitykseen liittyen. Tutkimuksen tarkoituksena oli saada tietoa turvallisesta ohjelmistokehityksestä ja selittää, miten turvallisuusominaisuuksia implementoidaan kehitettäviin järjestelmiin. Tutkimus toteutettiin laadullisella menetelmällä, suorittamalla puolistrukturoituja haastatteluja seitsemälle suomalaisen IT-organisaation edustajalle. Aineisto analysoitiin temaattisella analyysillä ja tutkimuksen teoreettisena taustana käytettiin Anthonyn (1964) organisaation päätöksentekoprosessin viitekehystä. Tutkimus osoittaa, että turvallisen ohjelmistokehityksen eri vaiheissa toteutetaan erilaisia turvallisuusominaisuuksia. Yhtiön strategisen tason turvallisuuslinjaukset muunnetaan teknisiksi ohjeistuksiksi, jotka ohjaavat arkkitehtuuripäätöksiä, vaatimussuunnittelua, tunnistautumismenetelmän valintaa ja komponenttien integrointia. Anthonyn teorian vastaisesti, kehittäjien vaikutus päätöksentekoon voi olla huomattava; kehittäjät voivat asiantuntijoina valvoa korkean tason teknisiä päätöksiä, kuten arkkitehtuurisia ratkaisuja. He saattavat myös käyttää valtaa saadakseen aikaan muutoksia tiimin käyttämän kehitysmenetelmän valintaan virallisen turvallisuuspolitiikan vastaisesti. Tutkimus osoittaa, että sääntely on yrityksille suurin täytetyn vaatimuksen lähde, ja valintaperusteet tunnistautumismenetelmälle ovat samanlaiset kuin mihin tahansa komponenttiin liittyvät perusteet. Lisäksi tutkimus osoittaa, että järjestelmien kehittäjät eivät ota vastuuta nykyisten tunnistautumiskäytäntöjen ylläpitämisestä, vaan katsovat sen johtuvan sääntelystä.

Avainsanat: Turvallinen ohjelmistokehitys, kyberturvallisuus, turvallisuusominaisuudet, loppukäyttäjän tunnistautumismenetelmä

Glossary

SRE	Secure requirements engineering
BoK	Body of Knowledge
IS	Information Systems
GDPR	General Data Protection Regulation
NFR	Non-functional requirement
SbD	Security by Design
PbD	Privacy by Design
SPL	Software product line
EA	Enterprise architecture
DevOps	Development and Operations
CI/CD	Continuous Integration and Continuous Deployment
COST	Component-off-the-shelf
OSS	Open-source software
DSS	Decision support systems
SDM	System development methods
IAM	Identity and access management
CIAM	Customer Identity and Access Management

List of Figures

List of Tables

Contents

1 Introduction

The gatekeeper of the secure use of the software or application is the end-user authentication. Relating to passwords, everyone has an opinion. For users, they are painful. In the earlier days, passwords as an authentication method were a valid solution. Users managed well with eight-character passwords and accounts were safe from brute force attacks. The number of accounts was also lower. Due to digitalization, the number of accounts as well as the requirements for secure passwords have increased significantly. The suggested number of characters has risen, the usage of special characters is forced, and periodical changes of passwords are mandatory. (Stajano, 2011; Still et al., 2017) This has led to the use of insecure password practices, such as password reuse (Woods & Siponen, 2018).

One of the reasons why passwords are the most common authentication method provided for users is due to organizational fondness. It is a feature that is easy to implement in systems, at a low cost. (Sims, 2020) The negative aspects relating to passwords, however, are disturbing. In 2022 alone, over 700 million passwords were leaked (Keary, 2023). In academia as well as in the industry it is well-known that people are the weakest link to both security and privacy (Belanger & Xu, 2015; Crossler et al., 2013). It is the user that represents the greatest threat exposing information and assets to loss (Willison & Warkentin, 2013).

Due to the mentioned negative aspects related to passwords as well as their implementation prevalence, it is interesting to know why passwords are chosen as an authentication method during software development. Velásquez et al. (2018) performed a systematic literature review regarding authentication methods as well as creating a framework for comparing and selecting authentication methods. This framework focuses on reviewing the system to which the authentication method is implemented, the system's requirements and the need for security and then proposed a solution. This was an improvement to a situation, where eight different decision-making frameworks existed each of them having a slightly different approach to the matter, such as cost (Altinkemer & Wang, 2011), web authentication (Bonneau et al., 2012), IP Multimedia systems (Eliasson et al., 2009), users (Forget et al., 2015), authorization (Guel, 2002), paid mobile devices solutions (Kim, 2017), and 2-factor authentication schemes (D. Wang et al., 2016).

Those approaches do not, however, take a stand relating to organizational aspects. Velásquez et al. (2020) argued, that it is the development team's responsibility to choose which authentication technique to implement to the system. Therefore, it is interesting to find out what affects the decision-making process of a development team. Development teams are units of software development organizations and therefore organizational decision-making processes must be investigated.

Anthony's (1964) theory on the organization's three-level decision-making processes has been used to describe software engineering projects' decision-making processes. (Aurum et al., 2006) The framework states organizational decision-making activities occur in three different levels, strategic, management and operational levels. The direction of the decision is from the top to down; the strategic level determines the company's objectives and management decides resources to achieve these objectives. The operational level's responsibility is to ensure the execution of the tasks within the resources allocated to them. (Anthony, 1964)

In the context of secure software development, the theory may not be fully applicable as such. The theory does not take into consideration software developers' position as an expert more than operational personnel. Modern software development is fast-paced and relies on developer knowledge and autonomy. Graziotin et al. (2015) research explains that developers are cognitive creatures with emotions. Moreover, developers, like any humans perform their daily tasks in teams with its, sometimes contradicting, coherence and rules. Power relations are part of human interaction.

According to Niazi et al. (2020), the outcome is costly and ineffective if security is considered and implemented during the last phases of software engineering. Therefore, secure software development considers security requirements at the early stage of the software development process. Secure requirements engineering (SRE) is helpful when seeking secure and cost-effective design, development, implementation, and maintenance. Security decisions are made at various levels of the organization during secure development engineering regarding applicable standards (Rindell et al., 2021) and regulation (Goel & Shawky, 2014), secure requirements engineering (Niazi et al., 2020) and architecture (Hong, 2006), as well as secure development methods (Myrbakken & Colomo-Palacios, 2017). During that process, the selection of different security features should be controlled at all levels and in line with strategic objectives.

Security related to software is a combination of two aspects: the combination of used security practices during the software development and the selection of security features. To understand how these processes happen in organizations we need to ask these questions:

Q1: How do different security practices emerge related to software development activities?

Q2: How do the organizational decision-making processes affect systems security features?

To find answers to these questions, it is important to understand what secure software development consists of and how the organizational decision-making process works. Chapter 2 focuses on secure software development and Chapter 3 on organization decision-making. In Chapter 4 methodology is presented and Chapter 5 shows the results. Chapter 6 analyses how the results are comparable to previous research and Chapter 7 sums up conclusions with implications both to research and practice as well as limitations of this research.

2 Security in software development

Security has several definitions relating to software development such as secure code, security engineering, and secure requirements engineering. Security is part of organizational culture (Prakash Attili et al., 2022), requirements engineering (Niazi et al., 2020), architecture (Hong, 2006), method (Akbar et al., 2022), as well as code (Hein & Saiedian, 2016). It is also a multi-disciplinary concept, when code vulnerability is related to computer science, development methods relate to information systems. Software development has certain linearity as well as a hierarchy. This chapter presents the main features of secure software development from regulation to code-line details.

To gather an understanding of research related to this study a literature review was performed. Levy & Ellis (2006) define literature review as a vital part of establishing the body of knowledge (BoK) including the summary of what is already known and where new research is needed. The process started by discovering what kind of language is used relating to studies in this area. This is also suggested by Levy & Ellis (2006), who calls it keyword search and suggests it to be the initial step of the IS literature review.

During the process with the most interesting papers during keyword search, a backward and forward search was performed, which is also suggested by (Levy & Ellis, 2006). Backward search, put forth by Webster & Watson (2002) is a technique for reviewing the references of interesting papers. Forward search is a method of reviewing articles that cite the current article. Those techniques were used with the most promising scholars as well; what the author had published before or after the article currently in hand. In conclusion, suitable keywords were discovered by conducting forward and backward searches for both publications as well as to scholars. The keywords selected for the literature review were “decision”, “selection”, "software development", "software engineering" and "system development". The usage of synonyms was intended due to the variety of definitions for certain features.

Once suitable keywords were found, the next step was the selection of publications. Due to the multi-disciplinary nature of this research, the amount of previous research is significant. According to (Levy & Ellis, 2006) the quality of the literature may also be varying. To ensure the reliability of the papers used, the publication approved to literature review needed to have a minimum of Jufo 2 grading. Since the topic is related to information systems, computer science, as well as information security, the decision was made within the top publications in those fields (see Appendix B). When both the suitable keywords as well as publications were identified, the query string was ready to be filled into Scopus. Scopus is a citation and abstract database that searches for example from these databases related to computer science and information systems: IEEE Xplore, ACM, ScienceDirect and Springer.

The search resulted in 168 articles. After evaluating the suitability of each article, 30 were selected to be used in the initial literature review. The next step of the literature review was the processing to turn the data into information (Levy & Ellis, 2006). Atlas.ti software was used to analyse the literature. Based on the analysis, an overview of the secure software development process was gained. In the following parts is presented why security must be implemented in software from the beginning. It is also presented how and why security must be implemented in different levels of software, from legislation, requirements engineering, architecture and development methods to a single component.

2.1 The overall picture

Neglecting security in software engineering has led to various negative consequences (Conklin & Dietrich, 2005) such as security vulnerabilities (Mouratidis et al., 2005). In the earlier days, security in software engineering was seen as an add-on feature of the late stage of development. According to Mostert & Von Solms (1994), this add-on view was ten times more expensive than security being engineered into the system from the beginning. The fact that developers did not get support for secure engineering, was likewise a problem in the 21st century (Zulkernine & Ahamed, 2006). Furthermore, the lack of standardized methodologies led to undocumented knowledge of secure engineering (De Win et al., 2008). The lack of documentation as an issue was also addressed by Uzunov et al. (2014) who stated that a methodological perspective is needed. Nowadays it is considered vital for software to perform as intended on every occasion, even when it faces malicious attacks (Bernsmed et al., 2022). According to Lee et al. (2002), for software to survive attacks, it must follow software lifecycle process standards. Security is seen as a lifecycle (Traore & Woungang, 2013) that starts at an early stage, during the design of the process and continues to the very end of the software's existence.

The foundation of security in any software is in regulatory means such as laws (Goel & Shawky, 2014), standards (Rindell et al., 2021), company policies (Von Solms et al., 2011) and protocols (Costa et al., 2023). Application may vary regarding the region and therefore security legislation and its effect on software development security is also location-based. In the EU area, for example, General Data Protection Regulation (GDPR) defines approved methods for personal information handling (European Union, 2016). Standards, on the other hand, may be international, such as ISO27001 which is the standard for information security management (ISO/IEC, n.d.), and therefore widely used in today's software development organizations. Organizations, being influenced by the surrounding restrictions, cannot isolate themselves from their impact and Mihaylov et al. (2016), state that applications need to be compliant with the relevant standards and practice must happen inside these mandatory security frames. The regulatory frames are the ground for requirements, which are a collection of all the features that the software contains.

2.2 Secure requirements engineering

Requirements engineering in general is a part of the software development process with various decisions. The decisions relate to, inter alia, choosing appropriate methods and tools for various activities, identifying stakeholders, assessing feasibility, validating, and prioritizing requirements. Moreover, a decision is made relating to determining which requirements to implement during release planning (Aurum & Wohlin, 2005). Requirements can be functional and relate to business rules, administrative functions, and external interfaces, or they may be non-functional which usually relates to quality. Examples of non-functional requirements (NFR:s) are for example how many seconds the page must load or what is the value for the up-time percentage. According to Werner et al. (2022) NFR:s are vital for organizations' success. Requirements engineering has multiple definitions; it can be seen as a design decision (Evans et al., 1997), a problem-solving activity (Aurum & Wohlin, 2003), or a semi-structured or unstructured complex decision-making process made by the stakeholders (Aurum & Wohlin, 2005). This is due to its nature, during the requirements engineering, representatives of different stakeholders gather to present their needs relating to the developed system. Furthermore, all requirements are not equally important and therefore design decisions are necessary.

Regarding requirements engineering, security is an aspect that needs to take into consideration as well. Relating to secure requirements engineering, various models are found (Niazi et al., 2020); Riaz et al. (2016) state that goals are the ground for security requirements. If the company is operating in an extremely tightly regulated business, such as in the financial sector, the security aspect needs to be taken into serious consideration from the start. Security by design (SbD) is a holistic approach where security is seen as default and secure component implementation is a goal from the beginning (Casola et al., 2020) and Chehrazi et al. (2016) argue that it has a positive impact on the project success. A similar paradigm exists for privacy, privacy by design (PbD) and Hadar et al. (2018) define it as a policy that guides developers to apply solutions where better protection for privacy is achieved.

Furthermore, tools for helping in facilitating compliance with regulation exist; Mellado et al. (2010) provide a framework for security requirements engineering by conforming to security standards. Their framework consists of a security requirement engineering process for software product line (SPL) driven by security standards such as ISO27001, a security reference meta-model to manage the variability of those SPL artefacts related to security requirements, and a tool which implements the meta-model. To implement security into the software, it must be taken into consideration before any architecture is designed. In other words, the starting point of the development process must be security-enhancing regulation. Only then the software can fulfil the various requirements which are imposed on it.

2.3 Software architecture

Requirements and architecture stages tend to be overlapping in some cases and are not separate processes. Whalen et al. (2016) state that by carefully paying attention to architecture as well as requirements, critical systems can survive even sophisticated attacks. Furthermore, security requirements are specified in enterprise architecture models as well as in business processes (San Martín et al., 2022). Enterprise Architecture (EA) builds transformative capabilities from information, processes, people, and technology (Band et al., 2014).

Because architectural models are passed to applications and software (Hong, 2006), the security aspect is vital in the architectural phase for it to be transferred to the product. Bass (2017) identifies architects as role models; they have a mandate to comply with technical requirements, without forgetting organizational or cultural aspects.

As mentioned, secure software has advantages such as cost savings and resilience against malicious attacks. A need for safer design has been a topic for a while (Villarroel et al., 2005). However, the architecture phase has its difficulties. Security may be categorized as an abstract or non-functional requirement, and Atlee et al. (2007) present the difficulties that development projects face regarding non-functional properties; abstraction challenges, model manipulation and management challenges (traceability) as well as educational issues. Unfortunately, Buyens et al. (2009) present a dilemma regarding the favour of security principles; when priority is given to security principles it may cause unwanted side effects to architectural qualities such as performance or maintainability. Therefore, architecture means also decision-making between different qualities at the expense of security.

2.4 Development methods

According to Iivari (1996), the adoption of a software development approach or method is normally an organizational decision. A waterfall paradigm is an example of the linear method, which is a process where phases follow one another in a controlled way. It begins with requirements and design, is followed by the development and after the testing phase the software is ready for production and maintenance (see Figure 1).

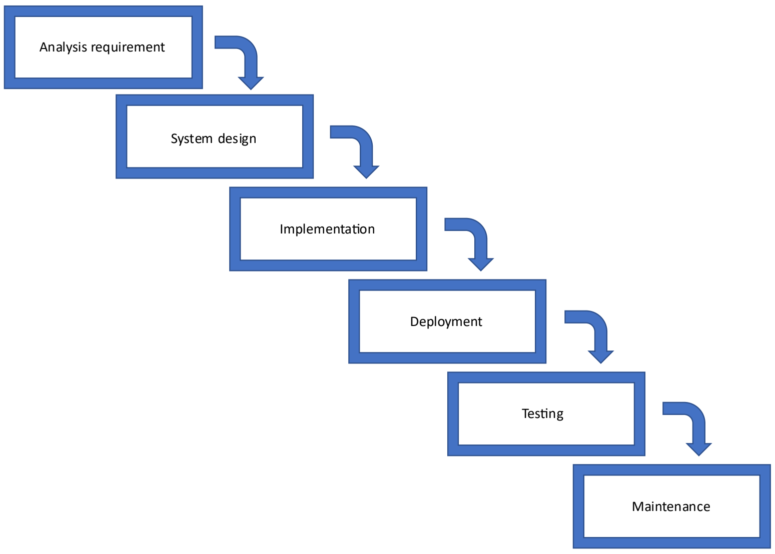

Figure 1. Waterfall model, after (*SDLC - Waterfall Model*, n.d.)

The waterfall model was widely in use but unfortunately had its issues. The strong dependence on the linearity of the process and compliance with the plan made it difficult and costly to make any changes during the development process. Issues relating to identifying customer needs were also significant and the result of the project did not meet customer needs. Many of the software development projects ended up in failure and therefore Beck et al., (2001) took a stand with Agile Manifesto. The manifesto underlined the following aspects of the software development process: “Interaction over processes, working software over comprehensive documentation, customer collaboration over contract negotiation and responding to change over following a plan.” After that software development has transformed into a more agile direction (Conboy, 2009). Next, some of the modern development methods used in software development are presented.

Development and Operations (DevOps) is a method, in which principles are continuous integration and continuous deployment (CI/CD). Figure 2 presents the process flow of the method. By making the collaboration between development and operations more effective, the method is aiming to improve the delivery of business value. This is done by increasing the automation of development, testing as well as administration. According to (Díaz et al., 2021) its popularity is enhanced by the speed it is providing for development, testing, and launch.

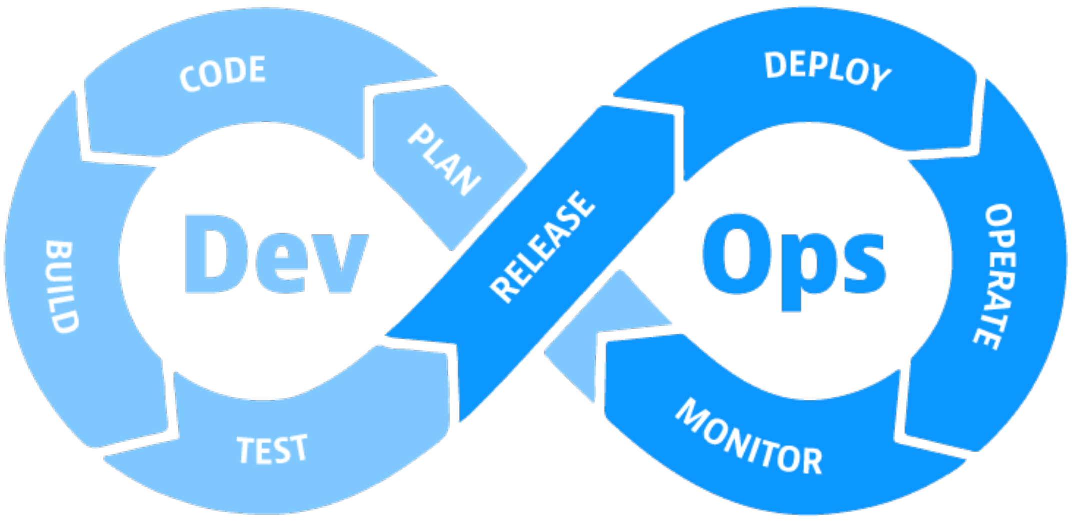

Figure 2. DevOps model (Gunja, 2022)

The demand for rapid delivery presents, however, new challenges. The challenges are related to security; how to ensure secure development while maintaining rapid development of DevOps. To tackle the problems related to the method's security, a new method was created based on it. DevSecOps is aiming to integrate security management throughout the whole process of development as well as during the maintenance phase. (Akbar et al., 2022) This is done by adding security controls and practices into the cycle of DevOps (Myrbakken & Colomo-Palacios, 2017).

Unfortunately, DevSecOps has its challenges; according to Rajapakse et al. (2022), problems with DevSecOps are related to four areas. The first is related to tools; usage of the development tools may be miscellaneous without a consistent method of use. The second challenge is related to practice such as the inability to automate the manual security practises to integrate it DevSecOps which emphasizes automation. The third challenge is related to infrastructure; the development environments are resource constrained and highly regulated and the cloud environments are complex for DevSecOps adoption.

In post-COVID-19 life, ways of working have changed and development teams have a lot of authority over their work. (Cucolaş & Russo, 2023; Spiegler et al., 2021) Therefore, developers' perceptions regarding security are important. Two viewpoints relate to developers and security; according to Lester (2010), one challenge lies in teaching since security is not integrated into teaching when learning programming principles. Moreover, developers' security and privacy perceptions have an impact on the daily development activities and in the end, the whole system (Hadar et al. 2018). Developers' perceptions towards security and privacy have been a topic for a while and some work is left to be done relating to app developers (Balebako et al., 2014; Linden et al., 2020), in engineering privacy (Hadar et al., 2018; Spiekermann & Faith Cranor, 2009) and organizational support toward developers (Assal & Chiasson, 2019).

2.5 Software security

Code reuse means using a standard application already existing as a solution to a certain function. This was originally introduced in 1968. It was the solution to emerged issues with software engineering at the time - how to build reliable, large systems in a cost-effective and controlled way. (Krueger, 1992) Back then, McIlroy (1968) suggested component libraries can be used in I/O conversion, dynamic storage allocation, numerical computation, and text processing. Nowadays, component reuse and libraries are common in the industry and Badampudi et al. (2016) define four different types of component origins: In-house developed components, Components-off-the-shelf (COTS), Open-source Software (OSS), and Outsourced components. Component reuse has its issues, as Biggerstaff & Richter (1987) point out: general technologies serve lower payoff, larger technologies have lower reuse potential, and it takes time before investments start to pay off significantly. According to Crnkovic (2003), however, components have the following advantages: they perform well, inter alia, for increasing productivity, managing complexity effectively, and creating a greater degree of consistency.

According to Li et al. (2006), business environment aspects such as critical time to market - demand affect which type of component is chosen. Moreover, according to Chatzipetrou et al., (2020) functional quality of the component influences its selection. Mehlawat et al. (2020) remind the importance of vendor reputation when selecting the component.

However, also a great risk lies in the components. According to Khan & Han (2006) danger lies for example in the use of the same component in different environments, it is secure to use in the first, but not in the other. This happens due to the diversity of usage contexts and security requirements. Testing is the manner of improving both code-line and components security; Hein & Saiedian (2016) mention static and penetration testing to improve the quality of components' security. Shahmehri et al. (2012) offer a passive testing method. Their method takes into consideration the issues with testing tools, such as their specification to only certain vulnerability and their possible out-of-date situations, which is a problem according to Austin & Williams (2011). It can be stated that different testing tools are efficient methods of catching vulnerabilities both in code lines and in the components.

Secure software production is vital, and the gatekeeper of the secure use of the software is authentication. It is a process, in which the identity of an entity, such as a user or device is verified (O'Gorman, 2003; Velásquez, 2021). It is a fundamental element in any security structure, as it is a prerequisite for entering the system and therefore the first line of defence against unauthorized access (O'Gorman, 2003; Velásquez et al., 2018a). User authentication factors are divided into three sections; what the person knows, what the person has, and what the person is (Choi & Zage, 2012). What a person knows, also named a knowledge factor is for example passwords, what a person has, also named the possession factor is, for example, smart cards and what a person is, the inherence factor is for example fingerprint (Choi & Zage, 2012; Velásquez et al., 2018).

Chang & Wu (1991) combined knowledge-based and possession-based factors and were the first to introduce 2-factor authentication for enhancing security (C. Wang et al., 2017). In general, combining several factors into a single authentication technique is called multi-factor authentication (O'Gorman, 2003; Velásquez et al., 2018a). A common example of a 2-

factor authentication method is SMS verification during login. To log in, the user must enter a 6-digit code they receive with a text message.

Overall, there are variations in practice related to secure software development; security can be seen as an add-on feature implemented at the final stage of the process. Nowadays, however, the modern development process should consider security from the start of the process. Even if the execution models of software development encourage continuous development, testing and deployment, secure software processes have some linearity. Requirements engineering as well as other design activities are usually at the beginning of the process. Overall, secure software development is a process where using different secure development practices, security is implemented in the software. The process also involves decision-making at different organizational levels, due to the different nature of security feature selection.

3 Organizational decision-making processes in software development

In addition to different software development activities categorization, software development organizations make decisions on three business levels that are strategic, tactical, and operational (Nguyen, 2006). According to Anthony (1964), organizations' decision-making is based on the purpose of management activities of three types: strategic planning, management control and technical control. The difference in decision-making is divided into four categories; the difference in the levels of the activities ("Does this affect to software library in use, or the total competence we have in our company?") the timeframe for the decisions ("How far-reaching this decision will be?"), the level of judgement ("Who makes this decision?") and the impact ("Does this only affect the development team or whole staff?") of a single action. Also, Shanks et al. (2009) used it for the classification of employee cohorts relating to CRM System benefits. Next, different software development-related decisions are divided according to Anthony's framework.

3.1 Strategic decisions

Strategic planning includes decision-making activities relating to the organization's objectives. The organization may need to adjust in response to the impact that the change in the business environment has on it. An example of the recent change is, for example, mergers, takeovers, business model changes and new technologies e.g., cloud computing and artificial intelligence. (Tamm et al., 2022) Regarding secure software development, change may be for example enhanced regulation, after which decisions related to resourcing for attaining new objectives are needed as well as policies to govern them (Anthony, 1964). In that situation, for example, the organization needs to make sure they have adequate knowledge for creating needed changes to existing security policies. Strategic decision-making at the enterprise level involves determining the security budget by weighing the cost of security expenditures against alternative non-security expenditures. The answer to the matter depends on the decision-makers risk tolerance and the information available to them. (Anderson & Choobineh, 2008)

Strategic decisions are also related to the image of the company. According to Hussain et al. (2022), neglect of human values such as privacy and honesty may lead to serious reputation damage and financial loss. The creation of organizational culture is likewise a strategic decision, Prakash Attili et al. (2022) argue it's one of the influencing factors in information privacy assimilations. Business considerations, such as positioning concerning clients and rapid reaction to clients' needs as well as the creation of business process models as a vital instrument in supporting requirements engineering are part of strategic-level decision-making as well (González Moyano et al., 2022).

Rindell et al. (2018) argue that external regulation and organizational security requirements are the ground for secure software engineering. They also state that it seems that the setting of the standards is more important than the follow-through. Strategic-level decisions related to software development, therefore, have an actual impact on product security. Strategic governance is executed relating to companies positioning in the software ecosystem that they belong to (Valença et al., 2018). This is related to, for example, sharing benefits among partners and attracting new players.

3.2 Management decisions

Management control makes sure that achieving the objectives that the strategic level has planned is efficient and effective with the given resources. Given resources are, for example, the number of experts in the team, the technology in hand and the timetable in which to finish the development process. The management works for the best interest of the company. (Anthony, 1964) The role of management decisions in software development companies is interesting since it has been estimated that more than half of IT projects fail. Failure tends to happen when assessment criteria such as timeliness of completion, staying on budget, meeting client's needs, and well-running team management, are not met. (Octavianus & Mursanto, 2019)

A recent development to a more agile direction in software development companies puts pressure on resources management. Very often the number of teams is high, and the amount of expertise is varying. In addition to that, the developer may be involved in more than one project, and the congruence between dependencies and the coordination of action is critical. (Dingsøyr et al., 2023) In fast-phase agile environments, communication between team members is often informal or is handled with instant messaging tools leading the knowledge of experts to stay undocumented and eventually lost. Therefore, it may be difficult for management to keep track of what kind of knowledge each project has. (Lima et al., 2019) According to Rindell et al. (2019), secure software development is accomplished by adding specific security engineering activities to the software development process. To ensure security in software development processes, management activities should include resourcing sufficient tools and frameworks for supporting project management and for example, Lopez & Sharp (2022) present such tools. Their framework sums up keyways in which individual developers form security responses to meet the demands of circumstances.

3.3 Operational decisions

According to Anthony (1964), operational-level decisions are called technical control. Technical control oversees the effective usage of the resources allocated to them. Development teams, groups of specialists who work together to accomplish their goals of building software need some coordination to ensure both the effectiveness of the team as well as the product quality (Koushik & Mookerjee, 1995). There are several viewpoints on the operational decision-making processes of software development. According to Nguyen (2006) an example of a software development project consists of four phases that are project definition, requirements, design, and implementation. These phases and their order are like the first steps of the waterfall development method model (see Figure 1). However, there are several problems related to these software development management practices such as unmatching the organization's structure and hierarchy, their process centricity at the expense of interfaces, a separation of quality modelling from process modelling, lack of suitable models, and the oversight of business goals. (Nguyen, 2006)

Nguyen (2006) suggests concepts that could be used in managing software development decision-making models to be mappability, accountability, interoperability, and controllability. In agile development, management issues are tackled with the Scrum Master who is supposed to empower the team lead itself (McAvoy & Butler, 2009; Spiegler et al., 2021).

Regarding the bias of time and cost underestimation, benefit overestimation, scope overload and over-requirement, an outside view may have a mitigating effect on projects (Shmueli et al., 2016). Also, Arnott (2006) observes the challenges that cognitive biases present in decision-making. Decision support systems (DSS) is one area of information systems (IS) dedicated to supporting and improving human decision-making. The objective of a DSS project is to improve the outcome for a manager making an important decision and the decision process in general.

The challenge in an agile software development project's decision-making lies in its character as an inter-related, chain of decisions where one decision leads to even more decisions, leading to a process that spans the whole project. Despite agile methods' positive aspects, the high level of empowerment may cause negative consequences such as group thinking. In their research, McAvoy & Butler (2009) found out the existence of an Abilene Paradox, a concept of decision-making where group decision-making leads to an outcome that no individual wanted, but they believed it would be something everybody else wanted. Therefore, they suggest some form of managerial role for the project manager of an agile team. (McAvoy & Butler, 2009)

Relating to security, the product manager's role is vital. Regardless of the well-known techniques for implementing secure and privacy-preserving software, competent software development teams still produce data breach software due to inadequate recourses and priority to the matter. (Weir et al., 2023) Teams try to do their best to perform with the recourses allocated to them and for example, time pressure influences software products. The research by Austin (2001) revealed that developers do care about the quality of the work but are afraid of missing deadlines that are dedicated to them. Furthermore, Skinner & Parrey (2019) have studied time pressure effects on decision-making wider in cybersecurity and found that the impact is adverse.

Weir et al. (2023) developed a tool for managing security experts in teams. Their solution presents ways to engage project managers relating security and privacy even if no expert in such area is present. Relating agile software development, Tøndel et al. (2022) describe how influencing the priority that security receives in development teams is best to do through interaction and providing support for teams.

3.4 Developers' influence on decisions

Anthony's decision framework emphasizes hierarchical decision-making, but it raises a question about the experts' role in the organization. The model presumes software developers are subject to working purely under the guidance of their supervisors. This contradicts highly Agile development methods but is also an interesting suggestion in general. Tversky & Kahneman (1974) have argued, that humans' decision-making processes have various biases that affect the results. In the IS literature, there is evidence that the decision-making processes are not purely hierarchical. Numerous biases affect decision-making in IS system development (Kirs et al., 2001). While not specifically about power, further studies advanced a growing argument that IT practice should not be seen as a mere technical activity and as argued by Goulielmos (2004), systems development methods (SDM) enactment can and should be understood as a complex social activity influenced by the organizational and institutional context in which it takes place (Rowlands & Kautz, 2022)

Graziotin et al. (2015) study explains that developers are cognitive creatures with emotions. Overall, developers have many roles; in addition to programming their daily tasks are, for example, system testing and operations engineering. The increase in responsibilities is due to the shift left paradigm that combines systems operations as configuration and security into software development. (Kam & Arcy, 2022) It is impossible to master everything; a study performed by Linden et al. (2020) concluded that developers' security perceptions have not changed enough and Xie et al. (2011) looked for reasons why developers make security errors. One explanation can be the heuristic-based decision-making process that developers use. According to Oliveira et al. (2014), developers solve problems without considering all the possible information available, in other words, security not being part of developers' mindset while coding.

However, not all poor security behaviour is due to neglect of the matter. Studies show that developers come across interference that makes them choose not to implement security to software and when Assal & Chiasson (2018) studied what obstacles developers face during development processes regarding security, they found out that security and business contradict one another. This leads to the situation, that security practices must be violated to reach business goals. Another explanation for such an obstacle was, that no proper resources such as knowledge or expertise were available. Senarath & Arachchilage (2018) discovered that relating to implementing privacy to software, developers reported that it contradicts system requirements. According to Spiekermann et al. (2019), engineers face conflicts when performing according to the organization's policies such as lack of time or opposed design.

3.5 Secure software development decision-making

The organizational decision-making process and secure software development are also cross-sectional in certain respects as listed in Table 1 which presents a comprehensive overview of secure software development and decision-making. Themes that were suitable to both sections (software development and organizational level) are used as a ground for this research.

	Strategic level	Management level	Operational level
Security directives	Creation of policies. (Rindell et al., 2018)	Implementation of policies.	Execution of policies.
Secure architecture	Alignment with business processes. (San Martín et al., 2022)		Selection between contradicting qualities. (Buyens et al., 2009)
Development method	Adoption of the development method. (Iivari, 1996)		
Competence	What knowledge does an organization		

	want to retain and serve? (Razavian et al., 2023)		
Component selection	Component vendors' reputation consideration.(Mehlawat et al., 2020)	Components functional quality and reliability (Chatzipetrou et al., 2020)	Make or buy? (Borg et al., 2019)
Code quality	Establishment of an organizational system that engages in quality-oriented behaviour. (Ravichandran & Rai, 1999)	Allocating the development team's resources for secure code. (Weir et al., 2023)	Software vulnerabilities are blind spots in developers’ decision-making processes. (Oliveira et al., 2014)
Client requirements	Rapid response to customers' wishes consideration. (González Moyano et al., 2022)		
Privacy	Creation of organizational culture that supports the importance of privacy. (Prakash Attili et al., 2022)	The architectural decision between network centricity and identifiability of data. (Spiekermann & Faith Cranor, 2009)	Conflicting requirements of privacy and business. (Senarath & Arachchilage, 2018)
Authentication method selection	-	-	Technical suitability (Velásquez et al., 2018b)

Table 1. Framework for the research

Anthony’s theory implies that the impact of the decision is always top to down, the environment affects strategy, strategy to management and management to the operational level. However, according to Hekkala et al. (2022), the efficacy of hierarchical power is often overestimated. Of course, one characteristic of power is formal authority, but according to French & Raven (1959) four other forms of power are reward (the ability to bestow rewards), coercive (the ability to punish), expert (possession of skill or knowledge) and referent (from personal characteristics). Power is likely to be enacted by all kinds of actors, regardless of the official authority or hierarchical position (Markus & Bui, 2012; Newell & Marabelli, 2016).

Milne & Maiden (2011) explain power being relationships or dependencies between actors, and social entities. An actor may be an individual, such as a developer, or it may be a collective social unit, for example, a team. Friendship or association in an organization are examples of a relationship that ties these actors to one another. It is interesting to see, are the decision-making processes of software development organizations correspond with the presented framework.

4 Methodology

Topics related to security are part of software development; security features such as authentication methods are parts of the software and their selection has technical aspects, relating to for example integration. Even if the subject relating to organizational processes is not at the core of computer science (CS), this Thesis is aiming to provide new information to the field of CS. A suitable research method needs to be selected concerning the phenomenon under investigation. It imposes restrictions on the research method.

To investigate secure software development processes and decision-making, it is vital to understand its key features. Secure software development is a process that is guided by regulatory means and security has significance to several parts, requirements engineering, architecture, development, and integration, as well as code-line level. Decision-making processes in organizations are on the other hand hierarchical but have interesting exceptions that relate to power relations. According to Shah & Corley (2006), interpretive research is based on the belief that a deeper understanding of the phenomenon is possible only by understanding the interpretations of the phenomenon from those who have experienced it. As mentioned, it is not yet known what kind of reflection is behind decision-making processes so therefore a qualitative, interpretive method is selected for this research.

Furthermore, quantitative research such as descriptive studies usually answers “what” or “what sort of” questions and they aim for quantitative coverage or theory building (Tietoarkisto, n.d.) whereas the phenomenon under investigation is the ways how secure software development practises and decision-making processes emerge. Measuring such phenomenon with numbers requires a lot of work with definitions to do it successfully. We do not thoroughly understand the practice of secure software development and it may be too complex to understand with just surveys.

4.1 Semi-structured interviews

This research attempts to gain insight into the decision-making processes of software development organizations. When aiming to recognise new relationships, understand complicated processes and clarify the influence of social context, Shah & Corley (2006) suggests qualitative methods as a base for research. Qualitative methods can be used to present explanations and to build and test theory (Van Maanen, 1979) and help in developing an understanding of the convoluted phenomena from the perspective of those who are surrounded by it (Miles & Huberman, 1994; Shah & Corley, 2006).

This study is focusing on what individuals think relating to the subject so therefore the selected data collection method interviews. Myers (1997) suggests having a qualitative research approach when aiming to interpret socio-technical processes. For example, Aurum et al. (2006) used semi-structured interviews in their research regarding the decision-making processes of software engineering.

4.2 Interview questions

Theoretical background was used to help form the questions (see Table 1). Since the interview invitation was sent with a cover letter hoping to get participants from all levels of the organization, the questions were formed in such way that providing answers was not limited by the question. Also, since companies have multiple roles relating to software security such as producing their software, consulting other companies, or positioning as a buyer or seller, the form of the question was kept general. Overall, the interview consisted of nine questions, relating to own role in systems development, own competence, security directives guiding system development, development methods, technology commitment and components, software security, client security requirements, privacy, and end-user authentication method selection (See Appendix B).

To test the interview questions two test interviews were held for persons having a role in high-level decision-making. Based on the feedback some changes were made. The questions used with the test interviewees were not suitable for all roles of organizations and there was a need for more potential follow-up questions. After these changes, the interview frame was ready for the actual data collection.

4.3 Organizing the research

The participants were recruited with the help of a university project steering group. To understand how decision-making is different in separate levels of organizations, all kinds of experts were included in the invitation to the study. To take care of the ethical side of the research, the invitation was sent together with permission requesting to conduct research. The following information was provided to the participants in the study; the main topic of the study so that participants were aware of what the interview was about at a general level, a description of the research method, the practical way in which the research is completed, the length of the interviews, and how the material is processed.

Myers & Newman (2007) suggest a few guidelines for interviewers to follow. The first one is situating the researcher as an actor. In this study, the researcher's relationship with the interviewees was formed originally by the university project, the interviewer was a project worker, and the interviewees represented the steering group companies of the project. Other than that, the researcher introduced herself as a master's student at the University of Jyväskylä and the interviews were part of her Master's Thesis.

The second guideline is to minimize social dissonance (Myers & Newman, 2007) so that the interviewees would not feel uncomfortable. This was done by dropping some praise on some interesting answers. Some interviewees also used humour the researcher aimed to react to such with accurate feedback, laugh was part of the interview occasionally. The third guideline is the variety of voices to avoid elite bias. Elite bias means neglecting interview samples at various organizational levels. Table 2 lists the interviewees with their organizational positions. Unfortunately, the operational level of organization decision-makers is missing in the sample even though in the invitation the importance of the variety of representatives was addressed.

The fourth guideline is that everyone is an interpreter. This guideline recognises that subjects are creative interpreters of their worlds as we are of theirs. It took several analysing rounds with the transcript to gather what kind of reality the subjects meant. Fifth, the use of models such as mirroring is addressed to reduce imposing researchers' worldviews on subjects. This was done with the general form of the questions which did not use specific vocabulary but allowed the subjects to put reality into words. (Myers & Newman, 2007)

The sixth guideline is flexibility which means that the researcher has a minimal script and improvising is done during the whole interview. In many cases, the researcher focused on new information that came in the answers and asked more relating to the subject. Finally, the confidentiality of disclosures was taken care of. This was executed, for example, by explaining the purpose of the research and providing the transcript to those who wanted to check it. (Myers & Newman, 2007)

The interviews were held between the 23rd of February and the 16th of March with Zoom video length of the interviews about 30-45 minutes. The total number of interviews was 7.

	Position	Level	Company size	Company sector
1	Director	Strategic	500-999	Finance
2	Security specialist	Management	1000-	Software development
3	Strategic trainer	Strategic	1000-	Software and telecommunication development
4	Product manager	Management	1000-	Software development
5	Architect	Strategic	1000-	Finance
6	Architect	Strategic	1000-	Finance
7	Architect	Strategic	500-999	Finance

Table 2. Participants of the interview

According to Myers & Newman (2007), the interviewer should aim to build trust by introducing the research topic and describing the processes of pseudonymity of the data. Some participants of the research had not seen the research permission, since they were recruited by another person than the one primarily contacted. Therefore, the interviews started by going through the research permission. The interviewees were told that no personal information was to be handled. They were also explained that recordings will be transcribed, and the data will be pseudonymized. After transcription, the recordings will be deleted, and no individuals can be recognized from the data. Those who wanted to receive the transcriptions of their interviews as well as the abstract of the study for learning purposes for the organization.

4.4 Thematic analysis

The analysis method of this research is thematic analysis. According to Aronson (1995), the first steps of the thematic analysis are data collection and conversation transcription. Data collection was conducted through semi-structural interviews. Based on the nature of the researcher's interest, the details of transcriptions may vary from deep to light (Jordan & Henderson, 1995). Since the research is focusing on substance more than on the language used, less-detailed transcriptions were enough. After transcription, the next stage was the analysis.

The analysis started based on transcriptions with patterns of experiences. In the first stage, the patterns identified were "business", "life cycle", "stakeholder" and "technology". The second stage was to identify all the data related to the already classified patterns. After that came the cataloguing of related patterns into the sub-themes which are derived from patterns. Then came the argument building for theme choosing, which was done by referring to the literature. The argument building was conducted with the help of the framework for the research (see Table 1). After this process, the storyline was ready to be developed. (Aronson, 1995)

5 Results

This chapter first presents the overview of the results, one question at a time. After that, the interview data is presented in a table form (see Table 2) for analysing. This is to transform the data into a more understandable form.

5.1 Summary of the answers

Q1 How is your role related to system development?

All the participants were currently in managing positions in their companies at strategic and management levels. The positions can be divided into leadership and management positions. Leaders have supervision over teams and people whereas managers have high positions regarding technical decisions without direct supervisory responsibility. Leaders had a team of cybersecurity experts in their team and the leaders' daily job included supervising and supporting their performance. Managers had responsibilities relating to for example the selection of technologies in use or participating in a competitive tendering. Their tasks could include consulting teams of individuals but without supervisory positions. These participants were architects having wide responsibility areas from cybersecurity abilities management to transforming sanctions to technical requirements. Changes in the geopolitical situation concern the finance sector from the perspective of sanctions. Relating to authentication, Identity Access Management (IAM) and its successor Customer Identity and Access Management were mentioned as their responsibility (CIAM).

Q2 What kind of competence do you have related to systems security and how do you develop your competence?

Answers to this question can be divided into proactive and reactive approaches. Participants having a proactive approach invested their own time and money for training, courses, and conferences. They were puzzled towards persons working in the field of cyber security and not having a proactive view of learning.

Anyone who works in the field of information security or cybersecurity will be required to continue studying. You cannot be in this area unless you are prepared to constantly develop your skills, and study new things, because this sector is developing at a terrible pace and new technologies and vulnerabilities are constantly coming.

Reactive participants, on the other hand, learn through projects. The explanation presented to this approach was for example the trust towards experts, the lack of time but on the other hand the fast-evolving nature of technology.

I have made a conscious choice, even though I come from an expert background, and it is my responsibility to take care of the security of the entire organization to ensure that I give responsibility and allow the experts to perform their duties and not go micromanaging with my incomplete skills. You must trust experts.

Reactive learners gained competence in work life and from projects but also security training provided by employers. Furthermore, they presented having trust towards specialists that have the latest knowledge in the fast-evolving security field.

Q3 What kind of policies guide system development security?

Three important themes came up from these answers. Companies operating in a heavily regulated area mentioned law and legislation as being the first obeyed policy. In addition to those, companies would have internal policies for security and development, in order with their strategies or architectural guidance.

Since we operate in the financial business area, we are guided by even more external requirements than our internal development guidelines, especially when we talk about customer identification solutions.

Those companies not operating in regulated areas mentioned workshops with customer representatives where threat modelling is done. Also, a secure development life cycle was mentioned as an important policy.

However, all the participants mentioned cultural change and for example, shift left. The cultural change that the participants mentioned, was related to the idea of security being everybody's responsibility, and not something that someone else does. It was also addressed that everyone in the organization needs to understand the importance of security by themselves and have intrinsic motivation for security practices.

Well, in the big picture, of course, all people need to understand why security is important... It's not like something you must do because, for example, I say something because the management team or the board of directors say so. Or if there is a regulated sector so that regulation says [...] Of course regulation must and must be so when a compliant, but it is a bit of a bad motivator.

The importance of the cultural change was also due to both siloed expertise in teams and so-called happy path thinking, meaning the assumption of nothing bad will happen. Modern working life is heavily siloed, meaning that a single employee may have a very narrow job description and therefore a limited point of view regarding the big picture.

Human limitations, siloed expertise and... in my opinion and based on my personal experience, majority of the people do not question their line of thinking and meta-level in common [..] very common situation is that a concrete situation awakens a person to think "ahh I need to take into consideration this also" and that is, on the other hand, the ground of experience-based learning.

Shift left means considering security on the left side of the DevOps model (see Figure 2). It was widely addressed that policy written on paper is only that. "Papers are only papers." What happens in the practice, and how and when security is implemented is a lot more important.

Q4 How security affects different development methods?

Two main methods for development were mentioned. The first one was the waterfall method, which has its limitations relating to security. When security aspects are considered at the very end of the process, usually the testing phase, fixing security errors is slow and expensive. Participants described aspects of the waterfall model such as quality gates that the software needed to pass for the development to continue to the next phase of the process. These quality gate controls were performed by someone else than the development team.

A person familiar with the Lean world, Edward W. Deming has said that quality, as well as security, cannot be audited in the product at the end.

The waterfall model is rare in software development nowadays and more common are iterative methods such as DevOps. This, together with team autonomy was another major theme within the answers. Teams' autonomy and responsibility were mentioned several times. Developers have security checks that the software needs to pass, but no other person is going to check the quality of work anymore.

It is the responsibility of the developer to use the tools that exist and consult the experts. In the line of your job where you produce, you're in charge of the fact that your outcome is secure. This is a pretty big change from a waterfall model, where someone else always looks out for you and says, "Well done!". Today, the developer must think about it and bear responsibility for it.

Interestingly enough, the waterfall method is still somewhat in use within interviewed organizations. This was presented together with two kinds of viewpoints. The first was the developers' power usage who did not agree with management decisions related to development method selection. When a certain team worked with the same technology for 40 years in the waterfall method, the resistance to other methods was massive. The development team did not have to change their way of working and were allowed to continue working with the waterfall method. In other cases, the decision related to the development method was also up to the team to decide, if the job gets done. There was no developer resistance mentioned, just the team autonomy.

Regarding security practises, penetration testing was also mentioned as a way in which security is related to development activities. It was made clear, however, that ensuring the safety of a product at the end of the project is not cost-effective and security by design is a much more favourable approach.

Q5 To what kind of technologies organization has committed from the high level to the component level?

This was a question all participants could not elaborate on. Background or role in a strategic or governmental position was a reason for not participating in this kind of decision-making. In situations, where the company produces software for a buyer company, the technology commitment is a decision the buyer company does, not the producer.

Overall, variety in technology usage exists and transfer to the public cloud due to its superior security features is an active process in software companies nowadays. The transfer causes an interesting situation regarding team autonomy in the selection of the development method. The mandatory pieces of company training and the latest technical company policies created for developers to follow apply only to cloud services even if the development is done with other methods also.

In general, the variety in technology usage is due to some technologies being better suited to specific issues than others. For example, certain workflows are more suitable to build on Microsoft cloud service and if the company is producing something new from scratch, Amazon Web Service (AWS) cloud is a better environment for that development process. AWS was in use in other companies in the same business sector as well, with on-premises solutions, which means that the server rooms locate in the company's own facilities.

Relating to the future vision, the rise of container technology was recognized in several organizations in a positive light. Container technology was even mentioned as the most common technology to perform certain activities. Its benefits to the single developer were also mentioned; the developer does not need to spend time figuring out the details of the infrastructure in which the development is performed.

It is based largely on container technology, i.e. freeing the developer [...] does not need to think about what the infrastructure layer is, they just *take the productized container and start building the whole thing on top of it.*

Overall, the importance to offer relief to the developers with the improvement of software development environments was highlighted; when automating secure development practises, the single developer may focus on what they know the best which is programming.

Architectural guidance relating to technology usage is done by creating sets from which the developers can choose the used technology for each project. In practice, it is managed by blacklisting technologies which are expected to expire, or which usage made no economic sense. The guidance was also used if overlapping technologies were in use subconsciously. Team autonomy was mentioned here also, the company selects the supported technologies, and the team decides which suits the best.

Relating to component selection, in addition to security, its lifecycle is a key element; the technology needs to have a life cycle long enough and there must be experts for that technology in hand. Companies use different strategies to ensure the life cycle such as framework agreements from domestic software development companies. With contracts support for a certain technology or knowledge is provided for a specified period of time.

Component integration was a vital topic related to the matter as well. If a component is purchased outside of the company, integration is a process that needs to be handled carefully. Usually, the company itself takes care of the integration. An interesting viewpoint relating to component security was the fact, that self-produced software is not always safer. This can be seen as a summary of all the aspects related to the security of a component which is a combination of life cycle, knowledge, and integration.

Q6 How does your work relate to code line security improvement?

This was also a question all participants could not elaborate on for the same reason mentioned in Q5. Those, who were able to give insight, mentioned the selection of recourses and development of the processes related to the matter.

The resources helping the developers improve code-line security were several. The most mentioned resource was the selection of different security and automation tools. The tools were related for example to testing, such as static analysis tools of the code, or a tool that alerts if a code or a library has vulnerabilities on it. The decision related to the tool selection may, however, be influenced by developers since they have preferences relating to testing tools. A false positive is a result when a scanner does a test for a vulnerability and incorrectly finds it to be vulnerable (Rogers, 2008).

> *This like finding false positives is a pretty big deal in code analysis, that that's not worth it -- or you have to look at such a tool for that, so it doesn't bring a large number of false positives because those software developers don't like if they get a big number of false positives.*

An example of a resource was also a network of experts such as testers to be ready for consulting when projects are in a phase such expertise is needed. Another group of security experts that are found within organizations are security and privacy champions. These champions are individuals that have a pronounced interest or expertise towards security or privacy aspects of the software development process and product. Participants mentioned the importance of both identifying these individuals as well as promoting their existence to development teams.

Regarding supervision of development processes, a managerial viewpoint on projects, processes and teams was mentioned, as well as if some observations need to be further processed. One organization has been promoting the correlation between excessive threat modelling (at the beginning of the development process) and with decreased number of vulnerabilities (found at the end of the process in the testing phase). This was carried out with a ticketing system and is still an ongoing task.

Q7 How do customer's or buyer's security requirements effect system development?

For highly regulated organizations this part of the software development process is not that important since regulation and authorities are the main source of requirements. For example, in the finance and banking sector organizations produce the software for themselves and no buyer exists that could present requirements because they are derived from the legislation. Their customers, millions of end users were not seen in a position to make demands.

[--] that we cannot, we have over a million active customers and nearly 6 million passives, it is quite impossible in the sense that customers might think that they are an internal customer who then sets demands on ICT. It is a business where demands arise, and that also reflects the regulation.

However, when security requirements are received from the customer or the buyer of the software, they are transferred for example to technical product requirements. Security requirements also affect the development projects' workload and cost, increasing both.

Q8 How data privacy is meaningful in developed systems?

In general, privacy was seen as an extremely important factor in developed systems. It has been important before GDPR and at the latest, the Vastaamo security leak (Ralston, 2021) has brought the importance of privacy to the attention of all. In the highly regulated sectors, its importance as the most important regulative entity was repeated. ”The law that mandates us is stronger than the consumer's right”.

In the banking sector organizations need to balance the demand to know their customer and secure customer data in an accepted way. In other sectors, the practice is that if the software handles personal information, it needs to pass these privacy-specific checklists. After the beginning of the war in Europe in 2022, the location of the data is significant, as well as the location of the person having access to data. For software development projects, the processing of personal data means following certain measures and development teams are required to follow privacy-related checklists.

Overall, the approach to ever-increasing regulation relating to privacy such as GDPR was mixed: “This is how the law of regulation changes, but it's all right to keep us busy. “ Some were neutral about the matter and explained it to be “business as usual” whereas some had more negative perspectives and the time span of the implementation as well as the cost to businesses were highlighted.

The negative side goes on to say that with what timetable it has been put into practice and how rigorously it has been interpreted. The *implementation has been extremely expensive at a societal level, the costs it has entailed for companies, the cost of bringing old systems to pass regulation, is so unreal.*

EU regulation in general was seen as a positive factor but relating practical implementation of the data privacy regulation there was some dissatisfaction.

Q9 Have previous projects involved end-user authentication method selection, and if so, what were the aspects of it?

Overall, when discussing relating to authentication methods, the tripartition is between authenticating a system, an internal user, or an end user. The development of those different methods is handled separately and very often within different teams. The authentication of an internal user is critical for some roles, such as server administrator.

Some organizations do not have a selection process related to the authentication solution of the software, due to the conservative nature of the business. A username and password are an adequate solution for its use and the only one applicable due to the user interface restriction.

In other organizations, such as those that produce systems of critical infrastructure, the security and reliability of the solutions are extremely important. In the banking sector, where trust is the most important factor of the business, it is not possible to lose the trust of the customer otherwise the company is out of the business. Overall, companies operating in the banking sector have a special position relating the authentication. In the EU area, Finnish banks are in a unique position of providing domestic authentication services to digital services, even though it is not even the reason why they are in the business first place. It is not reasonable to go against the public expectations related to the authentication. Finance and banking sector companies need to be compliant with both security requirements as well as with relevant protocols such as OpenID Certified (OICD).

The authentication method is a component and one influencing aspect mentioned was the integration, whether the new solution is compatible with the rest of the system. Overall, the same criteria were applied to a large extent, such as those at a component level. The lifecycle of the technology at the technical level as well as at the expertise level was mentioned.

The expertise must be found that it would be like a terrible situation that we have such a state-of-the-art product in the background, but no one knows how to configure it. In other words, it must be either in-house or it is known that it is available from that kind of network of partners.

The solution provider itself came up in the answers also, from the point of view how trustworthy and known the partner is or what is the nature of the business partnership. Important partners are those, with who problems are solved and new solutions are developed. In some situations, there was a controlled process for the solution provider that starts with a request for information, after which a request for a proposal is presented to a smaller group of providers. In the end, one supplier is selected, and a partnership has been formed for producing the strong authentication service. As software development is a business, the cost of the solution is a meaningful aspect as well; too expensive solution without business advantage is not possible to implement without adaptation.

5.2 Synthesis

The interviewees presented different organizational levels and were able to describe the responsibilities of their role or position as well as those who or what they oversaw. The results provided by the interviews can be placed in the table form. Synthesis of the results is presented in Table 3 that is reflecting Table 1.

Overall, participants were able to give insight into strategic-level decisions relating to security features, as their role in their companies was at a high level. Topics such as secure architecture and privacy were features that the interviewees discussed widely. Surprisingly, management-level decisions were least mentioned, whereas operational decisions were described in more detail.

	Strategic level	Management level	Operational level
Security directives	Reaction to sanctions due to the geopolitical situation (P3, P6)	Implementation of general security policies into the technical policies (P6, P7).	Transformation to technical requirements (P7).
Secure architecture	What technologies the company has in use (P7), technology stack (P5), technology life cycle (P1, P6)		Selection of technical solution based on str. Level decision (P4).
Development method	Development methods are in line with business (P1)	Ensuring the needed knowledge and experts in teams. (P1, P5, P6)	Handling development method variation. (P1, P5, P6)
Competence	Organizational need for knowledge (P1)		Training for developers. (P6, P7)
Component selection	Component vendors reputation consideration (P1) component lifecycle (P1, P2)		Integration of a component. (P1)
Code quality	Cultural change towards developers' autonomy and responsibility to code quality (P5)	Allocating resources such as champions and automated tools for teams (P2, P5).	The decision of tools that improve code quality. (P2, P5, P7)

Client requirements		Interpreter between a client and a development team (P4).	
Privacy	Trust business forces to keep privacy at the centre of all operations (P1, P5, P6, P7).		Privacy controls in development (P2).
Authentication method selection	Lifecycle of the technology (P1)	Compliance with protocols (P7)	

Table 3. Synthesis of the results

6 Discussion

The purpose of this research was to investigate, how different security practices emerge related to software development activities and how the organizational decision-making processes affect systems security features. It was examined using a reference framework which was built on two aspects: secure software development and Anthony's (1964) analysis framework.

The answer to the first research question is threefold. Different security practices emerging may be divided into organizational-, technology-, and people-related matters. Regarding the organizational-related subjects, the creation of an organizational culture that supports and encourages taking security as a priority was widely addressed in the responses. This is in line with Prakash Attili et al. (2022) who mention that organizational culture is one of the influencing factors with a statistically significant influence on information privacy assimilation. More specific factors of organizational culture improvement were for example the shift left, which was identified as a rising theme in secure software development. Shift left is a paradigm that was noted also by Kam & Arcy (2022).

Part of organization-related practices of secure software development is also compliance with the regulation. Especially in highly regulated business sectors such as finance and banking the extensive organizational security requirements are guiding the data assets and software (Rindell et al., 2018). Companies representing such sectors referred to the external guidance of regulation several times.

Lastly, organizations operating secure software businesses need to reflect on their position with vendors and wider with the whole ecosystem of organizations. As Mehlawat et al. (2020) stated, vendor reputation is a business-level consideration and, in the interviews, similar themes were mentioned; vendors are partners and the commitment to the business partner is a commitment to also develop together and solve issues that possibly are emerging. Participants mentioned framework agreements but also the importance of partnerships with business partners to survive in the market and the ever-evolving environment. This is in line with Hussain et al. (2022) highlight the importance of reputation as one of the factors that may be damaged if organizations do not take human values such as privacy into consideration. Valença et al. (2018) state the importance of building a software ecosystem to share risks and costs and to gain business advantage.

Secondly, secure practices emerge through technology. The importance of comprehensive security and privacy perception is executed in security engineering with Security by Design (Casola et al., 2020; Chehrazi et al., 2016) and Privacy by Design (Hadar et al., 2018). These methodologies were mentioned among several participants. They highlighted the advantages of early adaptation of security such as lower cost, and higher quality. They also suggested that the same development team take care of both software development tasks as well as security considerations. According to the participants, no developer can outsource their responsibility for security in software. Rindell et al. (2019) call this security engineering when the software development process is augmented with special security engineering activities.

Moreover, technology-related practices consist of the life cycle of the technology as well. The security of a solution such as a component is greatly related to the support there is provided for it. Life cycles are long, from five years onwards and the companies need to be sure that the technology is maintained but also that a skilled workforce for the technology is available. Strategies, that companies use for this are, for example, the business contracts from domestic software companies providing knowledge for a certain technology.

In more detail level regarding technology, different testing practices are very general methods for detecting vulnerabilities in software. Academia presents various testing tools for components (Hein & Saiedian, 2016) as well as the possibilities of passive testing (Shahmehri et al., 2012) and the interviewees pointed out its importance also. Automated testing tools that spot vulnerabilities from code-line, components, libraries, as well as integrations, were mentioned in the interviews. Testing tools are valued by developers and according to Nguyen et al. (2022), developers wish that static analysis tools would help with security-related issues more than they currently do.

Thirdly, people are an important asset relating to the security practices of organizations. Software developers' knowledge and competence are vital assets for organizations (Lima et al., 2019) and therefore documentation has an important role in improving project teams' performance in the long run. This is because undocumented knowledge gets lost in the long run and no improvement takes place. Relating to the people, the participants introduced their organization's training practices. The trainings were for example a part of a transition to new cloud architecture and therefore mandatory. One organization allows its employees to use work time to pieces of voluntary training.

Developers' perceptions towards security were mentioned widely in the interviews. New development methods that highlight team autonomy make this topic interesting. Developers' privacy and security perceptions are broadly discussed topic and research shows it is not at an adequate level (Balebako et al., 2014; Linden et al., 2020). The reason presented by the participants, why developers are not interested in security was simply the lack of interest, but a deeper explanation of that gave more context. Among interview participants, the point of siloed expertise was raised; modern work life is going in a direction where individuals have a narrower scope of expertise, and it is a human quality to focus only on the specific area on hand. Security practices emerge through the combination of organization, technology, and people.

Relating to the organizational decision-making processes of secure software development, the answer to the second research question is twofold. A great amount of organizational decision-making processes was in line with the tripartition of strategy, management, and operations but there was also a great number of decisions affecting quite the opposite. Overall, the decision-making processes were compatible with the reference framework. Regarding the organization's culture, several participants mentioned its importance to secure software development processes. Furthermore, other strategic level considerations, such as knowledge and lifecycles of technologies were similar related to the framework and widely addressed with participants.

Interview participants representing strategic and management level roles made high-level organization-related decisions related to security. The participants described, for example regarding security architecture, strategic level decisions such as the selection of the technologies used in the company. The selection was related to for example how many technology ecosystems the company is providing to developers to use, or what technologies to cut off if overlapping technologies were in use. One big theme overall relating to technology selection was the life-cycle aspect. The technology to which the company is investing needs to be maintained in the future (in the next five years) and it is a strategic level reaction when a certain technology is reaching the end of its lifecycle. Also, the know-how for the use of technology was mentioned, regarding the high qualities of a certain technology there need to be experts who are capable of utilising the tech. Furthermore, the security directives such as security policies, and the establishment of such is a responsibility of the strategic level. This is in line with Rindell et al. (2018) who have argued that external regulation and organizational security requirements are the ground for secure software engineering.

Contradicting the theory, the impact of an action may occur oppositely. Overall, the influence of a single developer may be significant regarding decisions that are made in the companies. According to (Iivari, 1996) a development method selection is an organizational decision. As mentioned, different development method has a different viewpoint on security, if added at the end, the cost becomes higher and the quality lower. In the interviews, the importance of including security in the development from the start and therefore more agile method favouring was mentioned by several times. However, the same participants mentioned teams' autonomy in the selection of the development method. In practice, the developers wished to stay in a waterfall model, after producing code in that way for years, rather than more modern. One participant explained how the transformation to the DevOps method was attempted, but the execution was simply not successful.

In some cases, a non-agile development method selection was also part of a less turbulent process. This was due to the one major theme among the interviews which was the team and developer autonomy. The matter was presented in a way that autonomy was a tool of the cultural change towards security becoming everyone's responsibility. Some guidance, for example, related to business existed, but other than that it was highlighted how the method itself was not meaningful, if things get done. To support the cultural change and team autonomy, companies provided different recourses for developers to utilize such as testing tools and automated tools indicating vulnerabilities in the code while programming. The selection of those tools was something that the participants mentioned as their responsibility related to code-line security, but it was also something developers had an opinion on. Developers' disfavour towards false positives of testing tools influenced the tool selection.

The participants underlined the freedom of choice and cultural transformation. Participants mentioned several times the importance of understanding security as a cross-sectional quality and not an add-on feature. The freedom appeared in several ways; the usage of agile development methods, where teams have autonomy in their work. Freedom came with responsibility too; developers and teams are in charge of the implementation of security and privacy.

Overall, it is interesting how the participants, on the other hand, highlighted the importance of the company security policies but on the other hand, listened eagerly to developers' opinions. The problems related to the high level of empowerment, such as biased decision-making are identified (McAvoy & Butler, 2009). The reason for the position as decision-maker affecting a higher level may be due to their role as experts more than just operational workers. Developers' opinions towards some security practices such as threat modelling was presented and eventually the organization ended up reforming the concept to align with business processes.

Authentication method selection takes place very similar to component selections in common. The latest changes among the interviewed companies were the implementation of a strong authentication or switching to mobile authentication. In general, the solution needs to be compliant with applicable protocols. The domestic market relating to the authentication service is interesting in several ways. Authentication is a process that banks provide to their customers (Tsai et al., 2021), but in addition to that strong authentication to public services is also the responsibility of banks even though authentication is not even the reason why they are in business. Overall, the system developers entrusted the regulation of maintaining the current authentication practice rather than themselves as the provider of such service.

Interesting observations during the interviews were the variety of ways of empowering developers to take security into a priority. One participant mentioned an ongoing process of improving the use of their ticketing system to point to benefits related to in investing security. Their goal was to use the system to demonstrate how paying attention to the threat modelling at the beginning of the process the number of vulnerabilities spotted at the end of the production is lower. This kind of activity, where collected data regarding software development outcomes is used to demonstrate to the developers the benefits of early adaptation of security, is fascinating.

Another tactic used by the strategic level in highlighting the security for developers was autonomy. Companies ended up settling the case this way, by providing different resources such as experts and automated tools for developers for consultation. However, there may exist a contradiction between the goal of making developers take responsibility and the means by which developer teams are encouraged to seek consultation in security-related areas of development. The supportive mechanisms that are created to entrust individuals to take responsibility for security-related ideas may be, in fact, working against it. Developers recognize the importance of security around them but are not obliged to change or evolve their security perception other than it is someone else's business, such as a security champion or automated testing tools. Furthermore, relying on automated scanners has its risks, since the tools themselves may be outdated for the latest vulnerabilities (Austin & Williams, 2011).

One outsourcing of security considerations was the usage of security champions. Security champions are the single point of contact regarding the security of each team. They are the persons in development teams that are in contact with the security team and ask for guidance. Their role is related to improving communication (OWASP, 2022). Participants mentioned the importance of recognizing the security and privacy champions in teams and publicizing their existence in organizations. According to Alshaikh (2020), establishing a cyber security champion network was one of the key factors in three Australian organizations to improve cyber security culture. The problem regarding software development lies in the situation of the security champion's involvement; they are in contact with the security team asking for guidance. The need, however, comes from the software developers, whose perception towards security and privacy may not be in line with security management (Xie et al., 2011).

Developers tend to see security as an external aspect related to programming. A method used in one of the companies participating in the research may be advantageous: security-related issues, such as vulnerabilities are considered a bug, or an error in a code, which is primarily the developers' responsibility to fix.

7 Conclusion

The purpose of this thesis was to find out how system developers choose the authentication method on the developed systems. The authentication method is one component of the developed system, and the same decision-making processes occur in both. At a high level, the component needs to meet the regulatory requirements. A major factor is the life cycle of the technology to which companies have different strategies ensuring it, such as framework agreements with business partners.

Implications for research

The purpose of this study was to gain insight into secure software development in light of organizational decision-making theory (Anthony, 1964). In some respects, the results were in line with the theory. However, some exceptions arose. Developers' characteristics influenced for example working methods, which normally is a strategic level decision. Also, the management level was not always aware of certain details even when they can be seen as a responsibility of their position in the organization. Overall, the theory is describing an ideal situation of the decision-making processes. It does not take into consideration individuals' characteristics, or their personalities with its effect on other individuals. Regarding secure software development, individual developers' effect on the security of the software is notable, as security extends to a low level, up to code lines. Based on this research, organizational decision-making in secure software development processes may not be as straightforward as Anthony's theory suggests.

Implications for practice

Organizations should continue their work on organizational culture relating to security. Cultural change towards organizations where security is a priority in every role must be continued at the strategic level. Concepts such as Privacy by Design and Security by Design are approaches in the planning phase such as requirement engineering, that foster cost-efficient and quality-assuring implementation of the security to systems. Furthermore, security is achieved by continuing the empowerment of the developer teams relating to security and autonomy but with suitable methods; security is not an isolated area of the software but an integral part of its quality. One practical example of such empowerment is the vocabulary used among software developers where a security vulnerability is identified as a software bug.

Limitations

There are some limitations to this study. Firstly, this was a qualitative analysis. The purpose of this research was to gain insight into the decision-making processes of software development organizations and how security is perceived concerning development. Due to the interpretative nature of this study, quantitative conclusions are not possible to do.

Another limitation is related to the interview participants since no software developers or operational-level experts were interviewed. Therefore, this research does not include developers' viewpoints on the matters under examination. Further research needs to be done by interviewing software developers.

Bibliography

Akbar, M. A., Smolander, K., Mahmood, S., & Alsanad, A. (2022). Toward successful DevSecOps in software development organizations: A decision-making framework. *Information and Software Technology*, *147*. https://doi.org/10.1016/j.infsof.2022.106894

Alshaikh, M. (2020). Developing cybersecurity culture to influence employee behavior: A practice perspective. *Computers and Security*, *98*. https://doi.org/10.1016/J.COSE.2020.102003

Altinkemer, K., & Wang, T. (2011). Cost and benefit analysis of authentication systems. *Decision Support Systems*, *51*(3). https://doi.org/10.1016/j.dss.2011.01.005

Anderson, E. E., & Choobineh, J. (2008). Enterprise information security strategies. *Computers and Security*, *27*(1–2), 22–29. https://doi.org/10.1016/J.COSE.2008.03.002

Anthony, R. N. (1964). Framework for analysis. *Management Services: A Magazine of Planning, Systems, and Controls*, *1*(1). https://egrove.olemiss.edu/mgmtservicesAvailableat:https://egrove.olemiss.edu/mgmtservices/vol1/iss1/6

Arnott, D. (2006). Cognitive biases and decision support systems development: A design science approach. *Information Systems Journal*, *16*(1), 55–78. https://doi.org/10.1111/J.1365-2575.2006.00208.X

Aronson, J. (1995). A Pragmatic View of Thematic Analysis. *The Qualitative Report*, *2*(1), 1–3.

Assal, H., & Chiasson, S. (2018). Security in the software development lifecycle. *Proceedings of the 14th Symposium on Usable Privacy and Security, SOUPS 2018*.

Assal, H., & Chiasson, S. (2019). "Think secure from the beginning": A survey with software developers. *Conference on Human Factors in Computing Systems - Proceedings*. https://doi.org/10.1145/3290605.3300519

Atlee, J. M., Cheriton, D. R., France, R., Georg, G., Moreira, A., Rumpe, B., & Zschaler, S. (2007). Modeling in Software Engineering. *29th International Conference on Software Engineering*.

Aurum, A., & Wohlin, C. (2003). The fundamental nature of requirements engineering activities as a decision-making process. *Information and Software Technology*, *45*(14). https://doi.org/10.1016/S0950-5849(03)00096-X

Aurum, A., & Wohlin, C. (2005). Aligning requirements with business objectives: A framework for requirements engineering decisions. *Proceedings of Requirements Engineering Decision Support Workshop*. https://wohlin.eu/redecs05.pdf

Aurum, A., Wohlin, C., & Porter, A. (2006). Aligning software project decisions: A case study. *International Journal of Software Engineering and Knowledge Engineering*, *16*(6), 795–818. https://doi.org/10.1142/S0218194006003002

Austin, B., & Williams, L. (2011). One technique is not enough: A Comparison of Vulnerability Discovery Techniques. *2011 International Symposium on Empirical Software Engineering and Measurement*, 97–106. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6092558

Austin, R. D. (2001). The Effects of Time Pressure on Quality in Software Development: An Agency Model. *Information Systems Research*, *12*(2), 195–207. https://doi.org/10.1287/ISRE.12.2.195.9699

Badampudi, D., Wohlin, C., & Petersen, K. (2016). Software component decision-making: In-house, OSS, COTS or outsourcing - A systematic literature review. *Journal of Systems and Software*, *121*, 105–124. https://doi.org/10.1016/J.JSS.2016.07.027

Balebako, R., Marsh, A., Lin, J., Hong, J., & Faith Cranor, L. (2014). *The Privacy and Security Behaviors of Smartphone App Developers*. https://doi.org/10.14722/usec.2014.23006

Band, I., Engelsman, W., Feltus, C., Paredes, S. G., Hietala, J., Jonkers, H., & Massart, S. (2014). Modeling Enterprise Risk Management and Security with the ArchiMate Language. *Open Group*.

Bass, L. (2017). The Software Architect and DevOps. *IEEE Software*, *35*(1), 8–10. https://doi.org/10.1109/MS.2017.4541051

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development* .

Belanger, F., & Xu, H. (2015). The role of information systems research in shaping the future of information privacy. *Information Systems Journal*, *25*(6), 573–578. https://doi.org/10.1111/ISJ.12092

Bernsmed, K., Cruzes, D. S., Jaatun, M. G., & Iovan, M. (2022). Adopting threat modelling in agile software development projects. *Journal of Systems and Software*, *183*. https://doi.org/10.1016/J.JSS.2021.111090

Biggerstaff, T., & Richter, C. (1987). Reusability Framework, Assessment, and Directions. *IEEE Software*, *4*(2). https://doi.org/10.1109/MS.1987.230095

Bonneau, J., Herley, C., Van Oorschot, P. C., & Stajano, F. (2012). The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. *Proceedings - IEEE Symposium on Security and Privacy*, 553–567. https://doi.org/10.1109/SP.2012.44

Borg, M., Chatzipetrou, P., Wnuk, K., Alégroth, E., Gorschek, T., Papatheocharous, E., Shah, S. M. A., & Axelsson, J. (2019). Selecting component sourcing options: A survey of software engineering's broader make-or-buy decisions. *Information and Software Technology*, *112*, 18–34. https://doi.org/10.1016/J.INFSOF.2019.03.015

Buyens, K., Scandariato, R., & Joosen, W. (2009). Measuring the interplay of security principles in software architectures. *2009 3rd International Symposium on Empirical*

Software Engineering and Measurement, ESEM 2009, 554–563. https://doi.org/10.1109/ESEM.2009.5315968

Casola, V., De Benedictis, A., Rak, M., & Villano, U. (2020). A novel Security-by-Design methodology: Modeling and assessing security by SLAs with a quantitative approach. *The Journal of Systems and Software*, *163*, 110537. https://doi.org/10.1016/j.jss.2020.110537

Chatzipetrou, P., Papatheocharous, E., Wnuk, K., Borg, M., Alégroth, E., & Gorschek, T. (2020). Component attributes and their importance in decisions and component selection. *Software Quality Journal*, *28*, 567–593. https://doi.org/10.1007/s11219-019-09465-2

Chehrazi, G., Heimbach, I., & Hinz, O. (2016). The impact of security by design on the success of open source software. *24th European Conference on Information Systems, ECIS 2016*.

Choi, S., & Zage, D. (2012a). Addressing insider threat using "where you are" as fourth factor authentication. In *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*. https://doi.org/10.1109/CCST.2012.6393550

Choi, S., & Zage, D. (2012b). Addressing insider threat using &#x201C;where you are&#x201D; as fourth factor authentication. In *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*. https://doi.org/10.1109/CCST.2012.6393550

Conboy, K. (2009). Agility from First Principles: Reconstructing the Conceptof Agility in Information Systems Development. *Information Systems Research*, *20*(3), 329–354. https://doi.org/10.1287/isre.1090.0236

Conklin, A., & Dietrich, G. (2005). Secure software design principles: A systems approach. *Association for Information Systems - 11th Americas Conference on Information Systems, AMCIS 2005: A Conference on a Human Scale*, *4*.

Costa, G., Degano, P., Galletta, L., & Soderi, S. (2023). Formally verifying security protocols built on watermarking and jamming. *Computers and Security*, *128*. https://doi.org/10.1016/J.COSE.2023.103133

Crnkovic, I. (2003). Component-Based Software Engineering-New Challenges in Software Development. *Journal of Computing and Information Technology-CIT*, *11*, 151–161.

Crossler, R. E., Johnston, A. C., Lowry, P. B., Hu, Q., Warkentin, M., & Baskerville, R. (2013). Future directions for behavioral information security research. *Computers & Security*, *32*, 90–101. https://doi.org/10.1016/J.COSE.2012.09.010

Cucolaş, A. A., & Russo, D. (2023). The impact of working from home on the success of Scrum projects: A multi-method study. *Journal of Systems and Software*, *197*. https://doi.org/10.1016/J.JSS.2022.111562

De Win, B., Scandariato, R., Buyens, K., Grégoire, J., & Joosen, W. (2008). *On the secure software development process: CLASP, SDL and Touchpoints compared*. https://doi.org/10.1016/j.infsof.2008.01.010

Díaz, J., Daniel L ´ Opez-Fernández, ·, Fernández, F., ´ Erez, J. P., Angel, · ´, Gonz´gonzá-lez-Prieto, G., Murphy-Hill, E., López-Fernández, D., Pérez, J., & Angel González-Prieto, ´. (2021). Why are many businesses instilling a DevOps culture into their organization? *Empirical Software Engineering*, *26*(25). https://doi.org/10.1007/s10664-020-09919-3

Dingsøyr, T., Olav Bjørnson, F., Schrof, J., Sporsem, T., & Sporsem Torsporsem, T. (2023). A longitudinal explanatory case study of coordination in a very large development programme: the impact of transitioning from a first- to a second-generation large-scale agile development method. *Empirical Software Engineering*, *28*(1), 1–28. https://doi.org/10.1007/s10664-022-10230-6

Eliasson, C., Fiedler, M., & Jørstad, I. (2009). A criteria-based evaluation framework for authentication schemes in IMS. *Proceedings - International Conference on Availability, Reliability and Security, ARES 2009*, 865–869. https://doi.org/10.1109/ARES.2009.166

European Union. (2016). REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL. *Official Journal of the European Union*. 31/03/2023https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679

Evans, R., Park, S., & Alberts, H. (1997). Decisions not requirements decision-centered engineering of computer-based systems. *Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems*. https://doi.org/10.1109/ecbs.1997.581926

Forget, A., Chiasson, S., & Biddle, R. (2015). User-centred authentication feature framework. *Information and Computer Security*, *23*(5), 497–515. https://doi.org/10.1108/ICS-08-2014-0058

French, J. R. P., & Raven, B. (1959). The Bases of Social Power. In D. Cartwright (Ed.), *Studies in Social Power* (pp. 259–269).

Goel, S., & Shawky, H. A. (2014). The Impact of Federal and State Notification Laws on Security Breach Announcements. *Communications of the Association for Information Systems*, *34*(3), 37–50. https://doi.org/10.17705/1CAIS.03403

González Moyano, C., Pufahl, L., Weber, I., & Mendling, J. (2022). Uses of business process modeling in agile software development projects. *Information and Software Technology*, *152*. https://doi.org/10.1016/J.INFSOF.2022.107028

Graziotin, D., Wang, X., & Abrahamsson, P. (2015). How do you feel, developer? An explanatory theory of the impact of affects on programming performance. *PeerJ Computer Science*, *2015*(8). https://doi.org/10.7717/peerj-cs.18

Guel, M. D. (2002). A Framework for Choosing Your Next Generation Authentication/Authorization System. *Information Security Technical Report*, *7*(1), 63–78. https://doi.org/10.1016/S1363-4127(02)00107-3

Gunja, S. (2022, December 15). *What is DevOps? Unpacking the purpose and importance of an IT cultural revolution*. Dynatrace.

Hadar, I., Hasson, T., Ayalon, O., Toch, E., Birnhack, M., Sherman, S., Balissa, A., & Menzies, T. (2018). *Privacy by designers: software developers' privacy mindset*. *23*, 259–289. https://doi.org/10.1007/s10664-017-9517-1

Hein, D., & Saiedian, H. (2016). Predicting attack prone software components using repository mined change metrics. *ICISSP 2016 - Proceedings of the 2nd International Conference on Information Systems Security and Privacy*, 554–563. https://doi.org/10.5220/0005812905540563

Hekkala, R., Stein, M. K., & Sarker, S. (2022). Power and conflict in inter-organisational information systems development. *Information Systems Journal*, *32*(2), 440–468. https://doi.org/10.1111/ISJ.12335

Hong, S. (2006). The software architecture of a secure and efficient group key agreement protocol. *WMSCI 2006 - The 10th World Multi-Conference on Systemics, Cybernetics and Informatics, Jointly with the 12th International Conference on Information Systems Analysis and Synthesis, ISAS 2006 - Proc.*, *2*, 274–278.

Hussain, W., Shaihin, M., Hoda, R., Whittle, J., Perera, H., Nurwidyantoro, A., Shams, R. A., & Oliver, G. (2022). How Can Human Values be Addressed in Agile Methods? A Case Study on SAFe. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, *48*(12), 5158–5157. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9677969

Iivari, J. (1996). Why Are CASE Tools Not Used? *Communications of the ACM*, *39*(10). https://doi.org/10.1145/236156.236183

ISO/IEC. (n.d.). *ISO/IEC 27001 and related standards*. Retrieved March 31, 2023, from https://www.iso.org/isoiec-27001-information-security.html

Jordan, B., & Henderson, A. (1995). Interaction analysis: Foundations and practise. *Journal of the Learning Science*, *4*(1), 39–103.

Kam, H.-J., & Arcy, J. D. ’. (2022). A Role Theory Perspective: Will Shifting Left Become a Pain for Application Developers? *Proceedings of the 17th Pre-ICIS Workshop on Information Security and Privacy*.

Keary, T. (2023, March 14). *Report: Hackers leaked over 721 million passwords in 2022* . https://venturebeat.com/security/hackers-leaked-over-721-million-passwords-2022/

Khan, K. M., & Han, J. (2006). Assessing security properties of software components: A software engineer’s perspective. *Proceedings of the Australian Software Engineering Conference, ASWEC*, *2006*. https://doi.org/10.1109/ASWEC.2006.13

Kim, J. Y. (2017). Efficiency of Paid Authentication Methods for Mobile Devices. *Wireless Personal Communications*, *93*(2). https://doi.org/10.1007/s11277-016-3286-9

Kirs, P. J., Pflughoeft, K., & Kroeck, G. (2001). A process model cognitive biasing effects in information systems development and usage. *Information and Management*, *38*(3), 153–165. https://doi.org/10.1016/S0378-7206(00)00062-8

Koushik, M. V., & Mookerjee, V. S. (1995). Modeling coordination in software construction: An analytical approach. *Information Systems Research*, *6*(3), 220–254. https://doi.org/10.1287/ISRE.6.3.220

Krueger, C. W. (1992). Software Reuse. *ACM Computing Surveys (CSUR)*, *24*(2). https://doi.org/10.1145/130844.130856

Lee, Y., Lee, J., & Lee, Z. (2002). Integrating Software Lifecycle Process Standards with Security Engineering. *Computer & Security*, *21*(4), 345–355. http://www.mitre.org/support/swee/paers_html/

Lester, C. Y. (2010). Shifting the Paradigm: Training Undergraduate Students in Software Security. *Fourth International Conference on Emerging Security Information, Systems and Technologies*, 117–122. https://ieeexplore-ieee-org.ezproxy.jyu.fi/stamp/stamp.jsp?tp=&arnumber=5633658

Levy, Y., & Ellis, T. J. (2006). A systems approach to conduct an effective literature review in support of information systems research. *Informing Science*, *9*. https://doi.org/10.28945/479

Li, J., Conradi, R., Petter Slyngstad, O. N., Bunse, C., Torchiano, M., & Morisio, M. (2006). An Empirical Study on Decision Making in Off-The-Shelf Component-Based Development. *ICSE '06: Proceedings of the 28th International Conference on Software Engineering*, 897.

Lima, M., Ahmed, I., Conte, T., Nascimento, E., Oliveira, E., & Gadelha, B. (2019). Land of Lost Knowledge: An Initial Investigation into Projects Lost Knowledge. *International Symposium on Empirical Software Engineering and Measurement*, *2019-Septemer*. https://doi.org/10.1109/ESEM.2019.8870171

Linden, D. Van Der, Anthonysamy, P., Nuseibeh, B., Tun, T. T., Petre, M., Levine, M., Towse, J., & Rashid, A. (2020). Schrodinger's security: Opening the box on app developers' security rationale. *Proceedings - International Conference on Software Engineering*. https://doi.org/10.1145/3377811.3380394

Lopez, T., & Sharp, H. (2022). *Security Responses in Software Development*. https://doi.org/10.1145/3563211

Markus, M. L., & Bui, Q. "Neo." (2012). Going concerns: The governance of interorganizational coordination hubs. *Journal of Management Information Systems*, *28*(4), 163–198. https://doi.org/10.2753/MIS0742-1222280407

McAvoy, J., & Butler, T. (2009). The role of project management in ineffective decision making within agile software development projects. *European Journal of Information Systems*, *18*(4). https://doi.org/10.1057/ejis.2009.22

McIlroy, M. D. (1968). Mass-produced software components. *NATO Conference of Software Engineering*, *October 1968*.

Mehlawat, M. K., Gupta, P., & Mahajan, D. (2020). A multi-period multi-objective optimization framework for software enhancement and component evaluation, selection and integration. *Information Sciences*, *523*, 91–110. https://doi.org/10.1016/J.INS.2020.02.076

Mellado, D., Fernández-Medina, E., & Piattini, M. (2010). Security requirements engineering framework for software product lines. *Information and Software Technology*, *52*(10), 1094–1117. https://doi.org/10.1016/j.infsof.2010.05.007

Mihaylov, B., Onea, L., & Hansen, K. M. (2016). Architecture-based regulatory compliance argumentation. *The Journal of Systems & Software*, *119*, 1–30. https://doi.org/10.1016/j.jss.2016.04.057

Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. SAGE Publications. https://books.google.fi/books?hl=fi&lr=&id=U4lU_-wJ5QEC&oi=fnd&pg=PR12&ots=kFZIZGVUZO&sig=2y1hsBz6wnxHx-isi3bTM6EA4iCg&redir_esc=y#v=onepage&q&f=false

Milne, A., & Maiden, N. (2011). Power and politics in requirements engineering: A proposed research agenda. *Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference, RE 2011*. https://doi.org/10.1109/RE.2011.6051646

Mostert, D. N. J., & Von Solms, S. H. (1994). A methodology to include computer security, safety and resilience requirements as part of the user requirement. In *Computers & Security* (Vol. 13).

Mouratidis, H., Giorgini, P., & Manson, G. (2005). When security meets software engineering: a case of modelling secure information systems. *Information Systems*, *30*, 609–629. https://doi.org/10.1016/j.is.2004.06.002

Myers, M. D. (1997). Qualitative Research in Information Systems Article in MIS Quarterly. *Quarterly*, *21*(2). https://doi.org/10.2307/249422

Myers, M. D., & Newman, M. (2007). The qualitative interview in IS research: Examining the craft. *Information and Organization*, *17*(1), 2–26. https://doi.org/10.1016/J.INFOANDORG.2006.11.001

Myrbakken, H., & Colomo-Palacios, R. (2017). DevSecOps: A multivocal literature review. *Communications in Computer and Information Science*, *770*, 17–29. https://doi.org/10.1007/978-3-319-67383-7_2/COVER

Newell, S., & Marabelli, M. (2016). Knowledge mobilization in healthcare networks: The power of everyday practices. In J. Swan, S. Nicolini, & S. Newell (Eds.), *Mobilizing Knowledge in Healthcare: Challenges for Management and Organization* (pp. 132–150). Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198738237.003.0007

Nguyen, L., Do, Q., Wright, J. R., & Ali, K. (2022). Why Do Software Developers Use Static Analysis Tools? A User-Centered Study of Developer Needs and Motivations. *IEEE Transactions on Software Engineering*, *48*(3), 835–847. https://doi.org/10.1109/TSE.2020.3004525

Nguyen, T. (2006). A decision model for managing software development projects. *Information and Management*, *43*(1), 63–75. https://doi.org/10.1016/J.IM.2005.01.006

Niazi, M., Saeed, A. M., Alshayeb, M., Mahmood, S., & Zafar, S. (2020). A maturity model for secure requirements engineering. *Computers and Security*, *95*. https://doi.org/10.1016/J.COSE.2020.101852

Octavianus, R., & Mursanto, P. (2019). The analysis of critical success factor ranking for software development and implementation project using AHP. *2018 International Conference on Advanced Computer Science and Information Systems, ICACSIS 2018*, 313–318. https://doi.org/10.1109/ICACSIS.2018.8618147

O'Gorman, L. (2003). *Comparing Passwords, Tokens, and Biometrics for User Authentication*. https://doi.org/10.1109/JPROC.2003.819611

Oliveira, D., Rosenthal, M., Morin, N., Yeh, K. C., Cappos, J., & Zhuang, Y. (2014). It's the psychology stupid: How heuristics explain software vulnerabilities and how priming can illuminate developers blind spots. *ACM International Conference Proceeding Series*, *2014-December*(December). https://doi.org/10.1145/2664243.2664254

OWASP. (2022). *Security Champions*. Security Culture Contents. https://owasp.org/www-project-security-culture/v10/4-Security_Champions/

Prakash Attili, V. S., Mathew, S. K., & Sugumaran, V. (2022). Information Privacy Assimilation in IT Organizations. *Information Systems Frontiers*, *24*, 1497–1513. https://doi.org/10.1007/s10796-021-10158-0/Published

Rajapakse, R. N., Zahedi, M., Babar, M. A., & Shen, H. (2022). Challenges and solutions when adopting DevSecOps: A systematic review. *Information and Software Technology*, *141*. https://doi.org/10.1016/J.INFSOF.2021.106700

Ralston, W. (2021, May 4). *They Told Their Therapists Everything. Hackers Leaked It All.* WIRED.

Ravichandran, T., & Rai, A. (1999). Total Quality Management in Information Systems Development: Key Constructs and Relationships. *Journal of Management Information Systems*, *16*(3), 119–155. https://doi.org/10.1080/07421222.1999.11518259

Razavian, M., Paech, B., & Tang, A. (2023). The vision of on-demand architectural knowledge systems as a decision-making companion. *Journal of Systems and Software*, *198*. https://doi.org/10.1016/J.JSS.2022.111560

Riaz, M., Stallings, J., Singh, M. P., Slankas, J., & Williams, L. (2016). DIGS-A Framework for Discovering Goals for Security Requirements Engineering. *ESEM '16: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 1–10. https://doi.org/10.1145/2961111.2962599Keywords

Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2019). Fitting Security into Agile Software Development. *International Journal of Systems and Software Security and Protection*, *9*(1). https://doi.org/10.4018/ijsssp.2018010103

Rindell, K., Ruohonen, J., Holvitie, J., Hyrynsalmi, S., & Leppänen, V. (2021). Security in agile software development: A practitioner survey. *Information and Software Technology*, *131*. https://doi.org/10.1016/J.INFSOF.2020.106488

Rindell, K., Ruohonen, J., & Hyrynsalmi, S. (2018). *Surveying Secure Software Development Practices in Finland*. https://doi.org/10.1145/3230833.3233274

Rogers, R. (2008). *False positives* (R. Rogers, Ed.; 2nd ed.). Syngress. https://doi.org/https://doi.org/10.1016/B978-1-59749-208-9.00007-1.

Rowlands, B., & Kautz, K. (2022). Power relations inscribed in the enactment of systems development methods. *Information Systems Journal*, *32*(2). https://doi.org/10.1111/isj.12322

San Martín, L., Rodríguez, A., Caro, A., & Velásquez, I. (2022). Obtaining secure business process models from an enterprise architecture considering security requirements. *Business Process Management Journal*, *28*(1). https://doi.org/10.1108/BPMJ-01-2021-0025

SDLC - Waterfall Model. (n.d.). Retrieved May 6, 2023, from https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm

Senarath, A., & Arachchilage, N. A. G. (2018). *Why developers cannot embed privacy into software systems?* https://doi.org/10.1145/3210459.3210484

Shah, S. K., & Corley, K. G. (2006). Building Better Theory by Bridging the Quantitative–Qualitative Divide*. *Journal of Management Studies*, *43*(8), 1821–1835. https://doi.org/10.1111/J.1467-6486.2006.00662.X

Shahmehri, N., Mammar, A., Montes De Oca, E., Byers, D., Cavalli, A., Ardi, S., & Jimenez, W. (2012). *An advanced approach for modeling and detecting software vulnerabilities*. https://doi.org/10.1016/j.infsof.2012.03.004

Shanks, G., Jagielska, I., & Jayaganesh, M. (2009). A framework for understanding customer relationship management systems benefits. *Communications of the Association for Information Systems*, *25*(1). https://doi.org/10.17705/1cais.02526

Shmueli, O., Pliskin, N., & Fink, L. (2016). Can the outside-view approach improve planning decisions in software development projects? *Information Systems Journal*, *26*(4), 395–418. https://doi.org/10.1111/ISJ.12091

Sims, J. (2020, May 5). *Passwords: why do we still use them?* https://www.raconteur.net/technology/cybersecurity/passwords-authentication/

Skinner, G., & Parrey, B. (2019). A literature review on effects of time pressure on decision making in a cyber security context. *Journal of Physics: Conference Series*, *1195*(1). https://doi.org/10.1088/1742-6596/1195/1/012014

Spiegler, S. V, Heinecke, C., & Wagner, S. (2021). An empirical study on changing leadership in agile teams. *Empirical Software Engineering*, *26*(3). https://doi.org/10.1007/s10664-021-09949-5

Spiekermann, S., & Faith Cranor, L. (2009). Engineering Privacy. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, *35*(1). https://doi.org/10.1109/TSE.2008.88

Spiekermann, S., Korunovska, J., & Langheinrich, M. (2019). Inside the Organization: Why Privacy and Security Engineering Is a Challenge for Engineers. *Proceedings of the IEEE*, *107*(3), 600–615. https://doi.org/10.1109/JPROC.2018.2866769

Stajano, F. (2011). Pico: No more passwords! *Proc. Security Protocols Workshop 2011*. http://www.schneier.com/crypto-gram-

Still, J. D., Cain, A., & Schuster, D. (2017). Human-centered authentication guidelines. *Information and Computer Security*, *25*(4). https://doi.org/10.1108/ICS-04-2016-0034

Tamm, T., Seddon, P. B., & Shanks, G. (2022). How enterprise architecture leads to organisational benefits. *International Journal of Information Management*, *67*. https://doi.org/10.1016/j.ijinfomgt.2022.102554

Tietoarkisto. (n.d.). *Tutkimusprosessi*. Kvantitatiivinen Käsikirja. Retrieved May 14, 2023, from https://www.fsd.tuni.fi/fi/palvelut/menetelmaopetus/kvanti/tutkimus/prosessi/

Tøndel, I. A., Cruzes, D. S., Jaatun, M. G., & Sindre, G. (2022). Influencing the security prioritisation of an agile software development project. *Computers & Security*, *118*, 102744. https://doi.org/10.1016/J.COSE.2022.102744

Traore, I., & Woungang, I. (2013). Software Security Engineering–Part I: Security Requirements and Risk Analysis. *Software Development Techniques for Constructive Information Systems Design*, 221–255.

Tsai, C.-H., Su, P.-C., Tsai, C.-H., & Su, P.-C. (2021). *The application of multi-server authentication scheme in internet banking transaction environments*. *19*, 77–105. https://doi.org/10.1007/s10257-020-00481-5

Tversky, A., & Kahneman, D. (1974). *Judgment under Uncertainty: Heuristics and Biases*. http://ebookcentral.proquest.com/lib/jyvaskyla-ebooks/detail.action?docID=1876866.

Uzunov, A. V, Falkner, K., & Fernandez, E. B. (2014). *A comprehensive pattern-oriented approach to engineering security methodologies*. https://doi.org/10.1016/j.infsof.2014.09.001

Valença, G., Alves, C., & Jansen, S. (2018). Strategies for managing power relationships in software ecosystems. *Journal of Systems and Software*, *144*. https://doi.org/10.1016/j.jss.2018.07.036

Van Maanen, J. (1979). Reclaiming Qualitative Methods for Organizational Research: A Preface. *Administrative Science Quarterly*, *24*(4). https://doi.org/10.2307/2392358

Velásquez, I. (2021). Framework for the comparison and selection of schemes for Multi-factor authentication. In *CLEI Eletronic Journal (CLEIej)* (Vol. 24, Issue 1). https://doi.org/10.19153/cleiej.24.1.9%20%20

Velásquez, I., Caro, A., & Rodríguez, A. (2018a). Authentication schemes and methods: A systematic literature review. In *Information and Software Technology* (Vol. 94). https://doi.org/10.1016/j.infsof.2017.09.012

Velásquez, I., Caro, A., & Rodríguez, A. (2018b). Kontun: A Framework for recommendation of authentication schemes and methods. *Information and Software Technology*, *96*, 27–37. https://doi.org/10.1016/J.INFSOF.2017.11.004

Velásquez, I., Caro, A., & Rodríguez, A. (2020). Multifactor Authentication Methods: A Framework for Their Comparison and Selection. In *Computer and Network Security*. https://doi.org/10.5772/intechopen.89876

Villarroel, R., Fernández-Medina, E., & Piattini, M. (2005). Secure information systems development - a survey and comparison. *Computers & Security*, *24*, 308–321. https://doi.org/10.1016/j.cose.2004.09.011

Von Solms, R., Thomson, K. L., & Maninjwa, P. M. (2011). Information security governance control through comprehensive policy architectures. *2011 Information Security for South Africa - Proceedings of the ISSA 2011 Conference*. https://doi.org/10.1109/ISSA.2011.6027522

Wang, C., Wang, D., Xu, G., & Guo, Y. (2017). *A lightweight password-based authentication protocol using smart card*. https://doi.org/10.1002/dac.3336

Wang, D., Gu, Q., Cheng, H., & Wang, P. (2016). *The Request for Better Measurement: A Comparative Evaluation of Two-Factor Authentication Schemes* *. https://doi.org/10.1145/2897845.2897916

Webster, J., & Watson, R. T. (2002). *Analyzing the Past to Prepare for the Future: Writing a Literature Review on JSTOR*. MIS Quarterly. https://www.jstor.org/stable/4132319

Weir, C., Becker, I., & Blair, L. (2023). Incorporating software security: using developer workshops to engage product managers. *Empirical Software Engineering*, *28*(21). https://doi.org/10.1007/s10664-022-10252-0

Werner, C., Li, Z. Sh., Lowlind, D., Elazhary, O., Ernst, O., & Damian, D. (2022). Continuously Managing NFRs: Opportunities and Challenges in Practice. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, *48*(7). https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9380710

Whalen, M. W., Cofer, D., & Gacek, A. (2016). Requirements and architectures for secure vehicles. *IEEE Software*, *33*(4), 22–25. https://doi.org/10.1109/MS.2016.94

Willison, R., & Warkentin, M. (2013). Beyond deterrence. *MIS Quarterly*, *37*(1), 1–20. https://doi.org/10.25300/MISQ/2013/37.1.01

Woods, N., & Siponen, M. (2018). Too many passwords? How understanding our memory can increase password memorability. *International Journal of Human-Computer Studies*, *111*, 36–48. https://doi.org/10.1016/J.IJHCS.2017.11.002

Xie, J., Lipford, H. R., & Chu, B. (2011). Why do programmers make security errors? *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. https://doi.org/10.1109/VLHCC.2011.6070393

Zulkernine, M., & Ahamed, S. I. (2006). Software security engineering: Toward unifying software engineering and security engineering. In *Enterprise Information Systems Assurance and System Security: Managerial and Technical Issues*. https://doi.org/10.4018/978-1-59140-911-3.ch014

Appendices

A Journals of literature review

Empirical Software Engineering

IEEE Transactions on Software Engineering

Information and Software Technology

Applied Soft Computing

Information Sciences

Computers and Security

Journal of systems and software

IEEE Software

ACM Transactions on Information Systems

International Journal of Information Security

Information Systems

Computers and security

Information Systems Journal

Information Systems Research

B Interview questions

1. Describe how your work over the past five years has been linked to the development of systems.
    a. Subscriber
    b. Producer (technologies)
    c. Project administration
    d. Senior management/programme management
    e. Stage/part of the production chain
2. What kind of skills do you feel you have in relation to the security of your systems?
    a. How have you developed these skills?
    b. training
    c. other own learning
3. What kind of guidance has guided the safety of system development?
    a. Policies
    b. Existing tools
    c. Customer
    d. Who has developed them?
4. If you recall, with what methods of work have systems been developed in previous projects, how did they reflect security considerations?
    a. Whether certain development methods were used
    b. Were there safety-related policies/parts?
5. Is the organisation committed to specific technologies in the development of the system?
    a. What is the commitment? Flexible, tight?
    b. Will the system development be guided to the use of ready-made components?
    c. Do safety considerations affect which components are used?
    d. Commercial (certain infrastructures), open source, in-house?
6. Describe how your work relates to the details of the software safety improvement of the system being developed.

    a. Customer perspective: quality
    b. Code quality
    c. High-level quality programmes
7. How do the safety requirements of the customer/buyer guide the decision-making process of the system development?
    a. How are the requirements obtained?
    b. Organisation/subscribe/end user
8. Tell us how privacy issues of personal data are relevant to the systems being developed.
    a. Saving a lot of user data?
    b. Personal data or organisational data?
9. Have your previous projects been associated with the selection of end-user authentication methods?
    a. If a choice has been made, do you remember what affected it?
    b. And if you have not joined the decision-making process, why?