The Impact of Enterprise Culture on DevOps Adoption: A Study of Security and Collaborative Practices in Software Development

Gry Heidi Solli

Thesis submitted for the degree of
Master in Applied Computer and Information Technology - ACIT
(Cloud-based Services and Operations)
60 credits

Department of Computer Science
Faculty of Technology, Art and Design

Oslo Metropolitan University — OsloMet

Spring 2023

The Impact of Enterprise Culture on DevOps Adoption: A Study of Security and Collaborative Practices in Software Development

Gry Heidi Solli

The Impact of Enterprise Culture on DevOps Adoption: A Study of Security and Collaborative Practices in Software Development

http://www.oslomet.no/

Printed: Oslo Metropolitan University — OsloMet

Abstract

Through the years, the need for faster release of software product has increased. DevOps, a combination of software developers and IT operations collaborating together, includes using automation, container technology and cloud computing to ensure that software is developed, deployed, and monitored. The practices of fast software development in DevOps impacts the prioritisation of security. Finding the balance and time between creating and fixing code but also trying to integrate security can be a tremendous challenge. This thesis aims to find out how companies in Norway use DevOps and security for software development, how they collaborate and how their company culture impacts their work.

For this study, a mixed method of qualitative and quantitative approach was chosen to collect data for the research questions. All participants worked either by writing code or setting up environments for developing and testing code, or both.

There is no agreed standard definition of the term DevOps. Companies were in both low-, mid- and high level of DevOps. A mandatory DevOps course should be provided so students who enter the IT industry are aware and knowledgeable of what DevOps is. All informants agreed that security is important, and the majority agreed that everyone should have some responsibility regarding security. People come from different educational backgrounds, have different experiences and knowledge. Therefore, it should be a standard for all IT companies to at least offer a basic security course for their employees. Communication channels seem to be the key for most to establish collaboration between team members, multiple teams, projects and subject groups. Social arrangements are common to create and build connection.

To start adopting using DevOps, a foundation of security practices, shared knowledge and a willingness to take risks must be in place. Companies should take time to plan how they will adopt DevOps, which tools, resources and services match their needs. A culture where employees feel safe to make mistakes, ask for help and discuss is essential. Social arrangements and environments support building good, trusted and vulnerable relationships with colleagues. This type of culture would help adopting DevOps and the encouragement to continuously learn and improve in an industry abundant with knowledge and new technology.

Sammendrag

Gjennom årene har behovet for raskere utgivelse av programvare økt. DevOps, en kombinasjon av programvareutviklere og IT-operasjoner som samarbeider, inkluderer bruk av automatisering, container teknologi og skytjenester for å sikre at programvaren blir utviklet, utgitt og overvåket. Praksisene for rask programvareutvikling i DevOps påvirker prioriteringen av sikkerhet. Å finne balansen og tiden mellom å skrive og fikse kode, samtidig som man prøver å integrere sikkerhet, kan være en enorm utfordring. Denne oppgaven har som mål å finne ut hvordan selskaper i Norge bruker DevOps og sikkerhet for programvareutvikling, hvordan de samarbeider og hvordan deres bedriftskultur påvirker arbeidet deres.

For denne oppgaven ble en kombinering av kvalitative og kvantitative metoder valgt for å samle data til forskningsspørsmålene. Alle deltakerne jobbet enten med å skrive kode eller sette opp miljøer for utvikling og testing av kode, eller begge deler.

Det er ingen enighet om en standard definisjon av begrepet DevOps. Selskaper var på både lavt, middels og høyt nivå av å bruke DevOps. En obligatorisk DevOps-kurs bør tilbys, slik at studenter som går inn i IT-bransjen er klar over og forstår om hva DevOps er. Alle informantene var enige om at sikkerhet er viktig, og flertallet var enige om at alle burde ha ansvar for sikkerhet. Folk kommer fra forskjellige utdanningsbakgrunner, har forskjellige erfaringer og kunnskap. Derfor bør det være en standard for alle IT-selskaper å tilby et grunnleggende sikkerhetskurs for deres ansatte. Kommunikasjonskanaler ser ut til å være nøkkelen for de fleste for å etablere samarbeid mellom teammedlemmer, mellom flere team, prosjekter og faggrupper. Sosiale arrangementer er vanlige for å skape og bygge forhold.

For å begynne å ta i bruk DevOps må det være et grunnlag å praktisere sikkerhet, dele kunnskap og vilje til å ta risikoer. Selskaper bør ta seg tid til å planlegge hvordan de vil bruke DevOps, og hvilke verktøy, ressurser og tjenester som passer til deres behov. En kultur der ansatte føler seg trygge på å gjøre feil, be om hjelp og diskutere er nødvendig. Sosiale arrangementer og miljøer støtter opp med å bygge gode, tillitsfulle og sårbare forhold med kollegaer. Denne typen kultur vil hjelpe til å ta i bruk DevOps og oppmuntre til kontinuerlig læring og forbedring i en bransje som er rik på kunnskap og ny teknologi.

Preface

This thesis started out as a curiosity for security in DevOps. The curiosity evolved when discovering that there are more factors to software development and security that meets the eye. Upon learning more about DevOps during my master's program and how there are still challenges to implement security, I wanted to continue to research of how others found DevOps, security and collaboration to be in their work and the industry. My supervisors helped me shape the scope of my thesis as they saw where my curiosities and interests lay.

I hope this thesis will give you insight of how some of the software development practices are being done in Norway and the different views on DevOps, security, collaboration and culture that exist. Whether or not you share the same views or experiences, I hope you find enjoyment reading them and find the research's contributions interesting and helpful.

Acknowledgments

First, I want to thank my supervisors H. Haugerud, I. Hassan, and E. Socchi for their guidance, support and encouragement while working on my master's project. I am grateful for their supervision and advices they gave to help me accomplish my goals for this thesis.

Secondly, I want to thank my friends for their support, cheers, encouragement and belief to get my master's degree at Oslo Metropolitan University (OsloMet) during these past two years.

Thirdly, I want to thank everyone who were interested and took the time to participate in my master's project and shared their thoughts, experiences, opinions and knowledge of their work. This thesis would not be complete without your participation and contributions.

Finally, I want to thank my friends and professors for these past five years at OsloMet. You have thought me much through the years both in and outside of class. I will always have fond memories working and learning together at OsloMet's Datatorget.

Oslo, May 2023
Gry Heidi Solli

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Technology has been evolving at a fast-pace the past decades, creating competition between companies and a higher demand of quality software deployments. By using agile methods and DevOps, software development teams are able to deploy faster quality code, which previously would take weeks or months to deploy. Closer collaboration between developers and operations can only happen if their silos are removed. Although security was not previously highly prioritised in software development, there has been an increase of security awareness the past few years. Actions for implementing security in software development are currently taken more seriously due to previous warnings and experiences with security breaches which had serious consequences in various companies. Company culture impacts on the motivation to ensure high quality and security are upheld.

1.1 Motivation

DevOps, a combination of software developers and IT operations collaborating together, includes using automation, container technology and cloud computing to ensure that software is developed, deployed, and monitored. While focusing on continuously improving and learning together, the developers and operations can accomplish their phases in their collaboration, which creates what is known as continuous integration and continuous delivery/deployment (CI/CD).

By following agile methodology, software developers know how to break down their work into smaller tasks, and complete them in iterations (also known as sprints) until the software end product is completed and deployed. By following the DevOps practices, the team will continuously work on the software following CI/CD. Therefore, software developers continue to be involved with the software product after deployment, which they usually part ways from after concluding their work.

Raising security awareness shows the importance of integrating security as a natural part of DevOps, turning DevOps to DevSecOps. There are several tools, guidelines, and strategies to ensure better security for

software development. These include automated security scanning tools of code, threat modeling, risk assessments, frameworks and checklists such as OWASP Container Security Verification Standards (CSVS) [1], OWASP Application Security Verification Standard (ASVS)[2] and National Institute of Security and Technology (NIST)[3]. OWASP provides a list of OWASP Top 10[4] security vulnerabilities to be aware of. Security activities can be arranged to increase security awareness, interest, and knowledge. A security role can be assigned for a software developer, though some companies have a security team tasked to ensure security with the software developers.

Code change requirements can happen quickly when following CI/CD with DevOps. While DevOps has its benefits, companies can experience struggles when adopting the practices seeing as they may not know how to adapt their legacy systems to use with DevOps and how to select the right tools and resources. It can take more time than anticipated when trying to find and learn the right DevOps tools needed. For instance, from the type of automation tools when building and testing code to what type of cloud computing resources and services to use for operating and monitoring. At the same time, company cultures have to adjust their way of thinking and their way of working to establish a work environment where teams feel safe to work and collaborate [25].

The practices of fast software development in DevOps impacts the prioritisation of security. Finding the balance and time between creating and fixing code but also trying to integrate security can be a tremendous challenge. Therefore, software developers may be reluctant or have a lack of interest to change how they work just to implement security [4]. This can result in software developers being resistant to the security teams' involvement [21, 33]. Others will focus on delivering the minimum requirements before deadlines without implementing security measures [24]. Security tools can take time to integrate in the CI/CD pipeline, and developers also have to understand the output of the security scanning tools [6, 33].

In 2022, security awareness was raised due to the Ukraine-Russia war. Both the Norwegian Data Protection Authority [13] and the Norwegian National Security Authority [31] warned about expected increased risk of cyberattacks because of the war. On their website, NSM offers supporting tools consisting of basic principles of security and checklists of security improvements and updates to implement to prepare for possible

[1]OWASP Container Security Verification Standard: https://owasp.org/www-project-container-security-verification-standard/migrated$_{c}ontent$

[2]OWASP Application Security Verification Standard (ASVS): https://owasp.org/www-project-application-security-verification-standard/

[3]NIST National Checklist Program: https://csrc.nist.gov/projects/national-checklist-program

[4]OWASP Top 10: https://owasp.org/Top10/

cyberattacks. In 2023, NSM's and several other companies and business' web pages were attacked[5].

The practice of DevOps differs from company to company, as DevOps can be interpreted differently depending on the company's definition of the term [33]. Other companies find DevOps unclear of what it is about [21]. A company's work and security culture impact how software development has CI/CD with security integrated. This thesis aims to find out how companies in Norway use DevOps and security for software development, how they collaborate and how their company culture impacts their work. Data on software development practices and work culture can contribute to examine how software development is presently carried out in Norway. Such knowledge can raise awareness on how to practice DevOps and to establish a culture which benefits collaboration and their work.

While agile methods and DevOps are popular in software development, not every company doing software development use them. Previous studies have researched practices, security, collaboration and culture from the individual workers' view on software development. This thesis use previous research methods in a similar fashion as a foundation to conduct research and to collect data. Previous studies, such as [4, 6, 33] used a qualitative approach by conducting interviews and analysing their results. An interview guide, following ethical considerations as discussed in Section 3.8, was created and followed when interviewing the participants. A survey with follow-up questions in form of statements with likert scale were sent after the interviews, where the data was used as a supplement for the findings of the qualitative approach.

1.2 Problem Statement

Building upon previous research and gathering data directly from people working in the IT-industry, this thesis addresses the following research questions:

1. RQ1: How is DevOps and DevSecOps defined and in what degree do they practice it?

2. RQ2: To what extend is security important to their software development process?

3. RQ3: How do they collaborate together and what impact do their collaboration have?

4. RQ4: How does the company's culture influence their software development process?

[5]NSM about the attack: https://www.nrk.no/nyheter/nsm-om-dataangrep__vil − vurdere − ytterligere − tiltak − 1.16321079

RQ1 investigates how DevOps and DevSecOps are defined and practiced. It is interesting to see how their definitions are similar or different compared to each other. It is possible that their own definition is influenced by the degree they practice DevOps or what they have heard elsewhere. RQ2 investigates security awareness, interest and whether they are involved in security activities. RQ3 examines how they collaborate together to get software code out to production, as close collaboration is one of the main keys for practicing DevOps. RQ4 investigates how company culture creates and influences the work environment and practices for their employees. As such, the first three research questions together contribute to answer RQ4 while at the same time RQ4 contributes to answer theirs.

1.3 Thesis outline

After this chapter follows Chapter 2 where background and related works present previous literature and studies of DevOps and DevSecOps practices, tools and known challenges. Chapter 3 describes the methodology used to design and collect data. Chapter 4 presents the results of this thesis findings. Chapter 5 discusses the findings from this research and describes limitations and proposes future works. The final chapter, chapter 6, presents the conclusion. The rest of this thesis consists of the bibliography, the appendix for the interview guide and the appendix for the survey.

Chapter 2

Background and Related Work

This chapter will present background and related work for agile methods and DevOps in software development, the importance of security and collaboration, what tools and resources are typically used, and the need for cultural change. The chapter will also introduce the challenges companies face when adopting DevOps and integrating security.

2.1 Background

This section presents the background literature for DevOps, DevSecOps and challenges of adopting DevSecOps discovered by other researchers.

2.1.1 DevOps

Through the years, the need for faster release of software product has increased. A software product is created after a software development has been completed and released into production. Even after the release, the product still needs to be maintained and updated to prevent and fix issues that can occur. This is also known as the software development life cycle (SDLC).

DevOps stands for Development and Operations, where software developers are collaborating with the operations team to improve the SDLC and deliver software products faster. Developers write application code which goes through the CI phases. Operations receive the application code in the CD phases where the code is in the production environment. Figure 2.1, inspired by Octopus Deploy[1], shows the phases of CI/CD in DevOps. New tools and resources were created and made available to achieve the goals, such as automation, container technology and cloud solutions through configuration management. Taking advantage of these tools and resources is a part of making DevOps effective [5]. However, using DevOps is more than just using these tools and resources, as company culture plays a vital role to adopt DevOps.

[1]The inspiration for the DevOps infinity loop figure: https://octopus.com/devops/

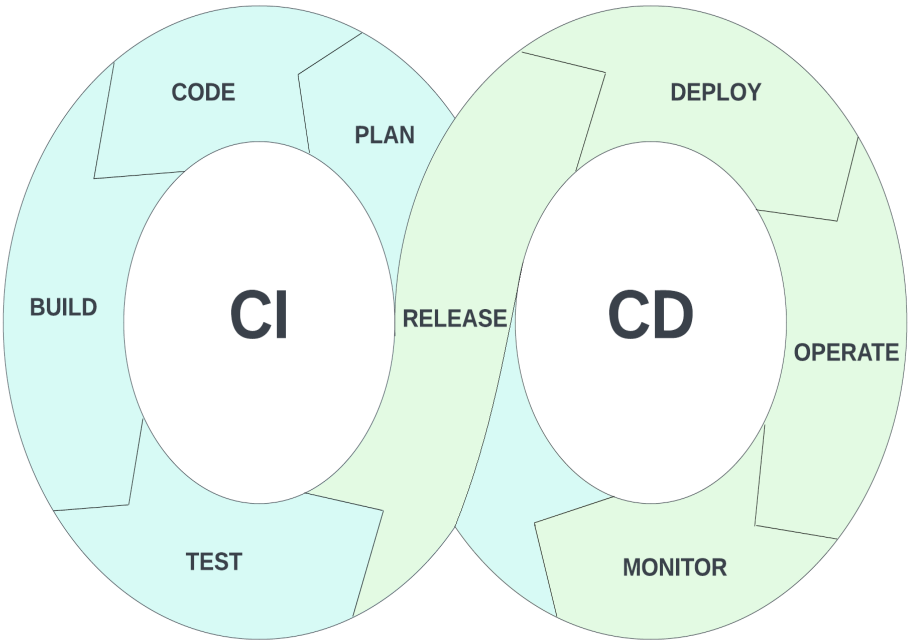

Figure 2.1: The DevOps infinity loop using CI/CD.

Agile methodology

Before DevOps was known as it is today, the majority focused on agile methodology. In [22], they tell the brief history that led up to DevOps. One of which how the the Agile Manifesto in 2001 introduced what is known as the values and principles of agile methodology [3, 19]. A key principle was to break up large releases for a software project into smaller batches to release incrementally. This was inspired by the theory of Lean Manufacturing, used by Toyota plants where they would use value stream mapping [22]. Agile methodology were introduce to make software development faster and easier. Software development teams break down their work into tasks which are managed with scheduled iterations (also called sprints). After an iteration is completed, the software development team has developed a part of the software product. The iterations continue until the end product is finished.

Scrum and Kanban are two examples of agile methods. Scrum uses iterations with the focus on the issues occurring during software development, where issues will be included in the iterations as tasks to be solved [30]. Kanban, on the other hand, focuses on implementing a visual workflow that represents the software development teams' tasks on a board. The board can either be physical or digital, where tasks can be moved towards the right as they progress and change status. While Scrum is more time-focused on the duration of each iterations, Kanban uses a less strict sched-

ule than Scrum. It is possible to combine both Scrum and Kanban, which is known as Scrumban.

In 2009, the presentation titled "10 Deployes Per Day"[2] by John Allspaw and Paul Hammond inspired Patrick Debois to create the first DevOpsDays where the term "DevOps" came to be [22]. Combining agile methodology and DevOps can give greater benefits, as DevOps support agile practices with creating better collaboration and common goals of producing quality code [2]. Still, it would take some time before the term was developed to what it is known and practiced as today.

Continuous Integration and Continuous Delivery/Deployment (CI/CD)

When talking about continuous integration and continuous delivery/deployment (CI/CD), it is sometimes interchanged with DevOps. This thesis refers CI/CD as part of DevOps. Continuous delivery is a ready made software build that can be deployed in any chosen time, while continuous deployment will deploy software builds automatically without human intervention [25].

Because the software developers are being included in all of the phases, they can get the feeling of ownership of the product they developed, which could affect how they work with the development. According to [6], a software developer feel ownership as it is their responsibility to fix bugs and again deploy code changes in production environment.

Automation

Automation in DevOps can remove a huge burden of manual work and save time. From [2] research's analysis, automation was found to be one of three benefits that stood out when combining agile methodology and DevOps. Automated tools can be integrated in the CI/CD pipeline, where building, testing and deploying code to production environment can be done automatically if the CI/CD pipeline is successful. With continuous deployment instead of delivery, it is possible to automatically deploy a software build without human interference. By using infrastructure as code (IaC), automation are used to speed up configuration management, provisioning and configuring systems with consistency [5]. Automation is not enough to practice high-level of DevOps, but it is a must to automate repetitive tasks [21]. It is also possible to integrate automated security tools in the CI/CD pipeline to catch security risks and vulnerabilities, which is explained in Section 2.1.3.

Container technology and cloud solutions

If the cloud is used, then DevOps should be practiced. An understanding of what the cloud provider's responsibility and what is one's own have to

[2] "10 Deployes Per Day": https://www.youtube.com/watch?v=LdOe18KhtT4

be clear so the responsibilities from the cloud provider are met [5].

The use of container technology has raised in popularity due to its light, easy and flexible use of running and testing software code, proving itself to improve the use in agile methods of fast development and delivery. The quick setup of containers save software developers time and are more lightweight than using virtual machines for the same purposes [7].

Another benefit is that container technology can be used as self-service, meaning that a software developer can independently build and start a container without any involvement of others [25]. When going through the small iterations, developers are able to integrate their code faster as it is easier to use test environments in containers before committing their finished code.

There are container platforms created for these purposes, Docker being among the most popular with how fast it can setup a lightweight environment [20] and widespread with its community and free options[3]. Seeing as working with container technology is increasing and cloud-based solutions are offered by both IT vendors and cloud providers, Open Container Initiative was created to establish standards for container image format and runtime [4].

Collaboration

Considering the importance of using resources and tools to improve software development, the company culture also plays a huge role as to how software development is done, separately and cooperatively. According to [21], to achieve quality code and fast deployment, DevOps teams need to utilise automation and cloud computing alongside with established communications between collaborating teams to reach high-level of DevOps. Communication was the second DevOps benefit that stood out from the findings by [2]. The findings from [27] made them agree with previous statements that culture indeed has to be changed to practice DevOps as DevOps tools are easy to learn and use. They believe that the challenges are not about the tools but about skills and culture.

Continuing from [21], they state that culture includes having all teams communicate internally and externally. Every team has a clear understanding of their own and other teams' responsibilities, and are able to communicate needs and share knowledge across all levels in the company. Company culture should make it common to use tools and resources such as automation and cloud-service to their advantage, so cognitive- and work load are reduced.

[3]Docker Hub: https://hub.docker.com/
[4]Open Container Initiative: https://opencontainers.org/

2.1.2 DevSecOps, implementing security in DevOps

There has been an increase of awareness and focus on security these past years, mostly due to the consequences of security breaches which could negatively affect the company and their customers. Security vulnerabilities can be discovered and exploited, giving attackers the access and opportunity to attack and obtain private and sensitive data. Malicious malware can be placed in third-party components undetected and take control over resources used in software development.

Security breaches can lead to businesses and organizations to shut down. Customers could have their private data leaked and suffer from the consequences of poor or lack of security policies and actions. To mitigate the chances of security breaches, security tools, guidelines and resources are available to increase security in software development. Focusing on security in DevOps, the term was changed to DevSecOps. To increase security, the collaboration in DevSecOps needs to include the security teams [5].

A framework was developed by [29] as a starting point to use existing practices to implement security activities. Their framework use five steps where the first two steps are an extension of agile practices such as stand-ups and weekly meetings, the next two steps to enforce risk mitigation tasks and knowledge transfer, while the last step is to report to stakeholders and management.

Container technology and security scanner analyzing tools

When using container technology, software developers have the option to use images created by other developers. Images contain the code which makes it possible to start and run containers, but it can take time to create images. The option of downloading and use existing images is practical for building containers, making the setup as fast as possible and ready to use. Unfortunately, using images from third-parties created by a non-confirmed developer leads to risks such as malicious code placed inside the images, giving hackers the opportunity to take over the container.

Docker has provided their own documentation regarding security awareness for their containers [17]. The open-source container runtime Kata was pointed out by [34] as an alternative container engine with better security isolation. In addition, there are security analyzing tools created to scan images and containers for security vulnerabilities. Using images not developed by the software teams themselves are considered to be vulnerable, and should not be trusted until they have been analysed and proven safe to use for running containers [20]. Therefore, analyzing containers might provide some security to help alleviate the security vulnerabilities.

Static and dynamic scanning are used to check images which create

containers to see if there are any vulnerabilities or malicious content. Static analysis scans images without executing the image's instructions to check them before using them, while dynamic analysis scans how the container itself behaves in an environment [7]. A scanning tool that performs a dynamic analysis will take longer time than performing a static analysis.

In [20], they present how the various type of static analysis tools display different type of results from checking vulnerabilities. Software developers have the option to choose the static analysis tool they need depending on the security measurements they want to see. It is possible to combine and use both static and dynamic security analysis, and use both of their output results for further analysis of security threats.

In [5], it is recommended to integrate automated static security analysis tools to the CI/CD pipeline to check for security vulnerabilities and security risks from third-party components. Static analysis will catch the common security issues and vulnerabilities when committing to the next phases of the pipeline.

Checklists

Using checklists is a way to ensure that security is applied by following the steps listed. As mentioned in Section 1.1, there are different types of guidelines to follow when creating a checklist, like NIST and OWASP CSVS.

In [1], a checklist was developed for improving Docker image security with the help of use cases. To make sure their checklist for the use cases were on par with the security standards of OWASP CSVS, the NIST guidelines were used to develop their checklist and to demonstrate how a checklist can be used for a software development project.

From their follow-up questions in [6], most of their interviewees knew of OWASP Top Ten list of vulnerabilities. It is possible that when checklists are used, some companies prefer to use the most common ones without acquiring further knowledge about lesser known ones. From the interviews in [4], the four organizations used checklists though they had different reasons why they found them useful.

Threat modeling

Threat modeling is a way for software development teams and security teams to assess what the security risks and vulnerabilities are for their software product. By doing threat modeling, designing the security plan can be done by thinking from an attacker's perspective [5]. One well-known and popular threat modeling framework is STRIDE, developed by Microsoft [34].

In [4], the practices of threat modeling and agile software development were identified in four Norwegian organisations. A good practice is to involve software developers during the threat modeling, while also supplement it with checklists. One of the best identified practices is to practice threat modeling regularly in intervals, as the participants of the study believed that it would make their product more secure.

Security Champion

In a software developer team, a member can have a security role, often called the Security Champion. As a Security Champion, their role is to make sure that the security standards are followed and practiced from the beginning to the end of the software development, helping the team to use and understand the security tools, and overall makes sure that the code is secure [28]. The Security Champion often has an interest in security and initiate both practice and discussion inside and outside of their teams [4].

2.1.3 Challenges and strategies of using DevSecOps

Many companies find it challenging when trying to implement security in DevOps. Teams will focus on delivering within deadlines rather than focusing on security, which only will lead to vulnerabilities and security risks for attackers to take advantage of [24]. While some identified challenges have proposed mitigation strategies, there can still be limitations of how far the companies manage to use these strategies.

Implement security early

As the integration of security becomes more important in the industry, there are beliefs that implementing security from the beginning of the CI/CD pipeline is crucial to make sure security in DevSecOps is prioritised and followed throughout the pipeline process [24]. Instead of waiting to implement security at the end before the software product release, implement automated security test cycles to the pipeline [5].

A software developer writes and commits their code, which triggers the CI/CD pipeline. The pipeline will give the developer feedback on the code whether it has successfully passed the pipeline or not. However, there should be security tools to scan the pipeline itself. If not, then undetected pipeline vulnerabilities can be taken advantage of by attackers, as one vulnerable part of the pipeline is enough for an attacker to take over the whole pipeline [34].

In [4], they were told that Microsoft Threat Modeling Tool was tried out, but was found to be too time-consuming for regular use and did not support third party components which made the tool unfit to use.

In [24], a multi-year software development project was followed. The project was well-funded and assembled by several vendors with their own teams. A goal for the software development project was that the vendors would start using agile methods and DevSecOps in the middle of the development which were already using the waterfall model approach. However, the vendors did not prioritise security due to their wish to reach the minimum requirements within the scheduled timetables. The researchers observing the project identified the security impacts of not including implementation of security, which could have been mitigated by incorporating DevSecOps from the beginning of the project and use available security tools to the pipeline for security improvement. Minimal effort from the vendors and their teams to implement security impacted the end product with increased vulnerabilities and security risks due to the sub-optimal efforts of DevSecOps. The well-funded software development project still needed security measurements, policies and teachings incorporated from the beginning to make sure security will be prioritised during the development of the project.

Choosing and understanding the security scanning tools

If container images are scanned and deemed secure before running the image to build a container, then it can be assumed that the container does not contain threatening security risks. However, if the scanner tools detect security risks within an image, there needs to be an evaluation to either try to fix the issue themselves or dismiss the image entirely.

Scanning containers is helpful to detect one of the major weaknesses of container security. Finding out what dependencies go together is a challenge in itself, but the dependencies also have to be checked for vulnerabilities and updated with their latest patch [5]. One of the huge risks is how containers containing privacy data and management can be exploited due to the direct access the containers have to the host's kernel [1, 34]. Attackers could be able to get access to one container which has access to the host's kernel, and benefit from its resources on the expense of the host.

Using a security analyzing tool is a risk in itself. The tools do not cover all the mistakes, and can also present false positives which only the developers with knowledge can recognises [33]. Therefore, a lack of knowledge can give a false sense of security because the tool's output was not understood correctly [6].

Cloud providers

A new challenge regarding cloud providers arose when Max Schrems brought to light how personal data between the Irish Facebook and Facebook Inc. in the USA were not protected well enough. The Experience report from Digdir, a yearly status of digitalisation in Norway, explained

the plan of action regarding the incident in 2020 [16]. Even if the personal data is stored within EU/EEA, the provider of the services outside of the EU/EEA is still outside of the EU/EEA law and may use the data as they please if there are no laws within their own country to prevent access to users' personal data. Taking this into consideration, actions have to be taken regarding how to handle personal data and security when it comes to cloud providers outside of the EU/EEA. The Norwegian Data Protection Authority has given guidance about Schrems II, including the two guidance documents from the European Data Protection Board to make sure that the laws are followed [14]. While popular cloud providers such as AWS and Azure are the dominating cloud providers in the market, users of their services still have to take into consideration how to protect their data when their data can be accessed by the service providers.

Collaborating with security teams

A security team assisting software developers to check their code can result to resistance from the developers. It was reported by an participant interviewed by [25] that they would try to avoid having the security team reviewing their code. The security team might comment on the developers' work and want them to change their code so the discovered security vulnerability is reduced or removed. The software developer might feel criticized about their work while they already feel pressure to focus on creating fast and quality functionality code and meeting deadlines.

Developers do not share the same security concerns as the security team. One of the reasons is how focusing on non-functionality such as security may not even be needed in the first place, as it is a security risk that might not happen [33]. Consequently, it can be easily brushed off as a task not needed to be prioritised. This can create a divide between the two teams, which will affect the overall working culture trying to implement security as the teams prioritise differently. A solution could be to have all developers acquire some of the required security knowledge for their work [33]. In this way, developers will cover the basic security tasks while also gain some understanding of what type of and why security is needed. Making security as a self-service could make the developers share the security goals with the security team while also being efficient without delaying their work as security is made faster with knowledge and automation tools [5].

If someone takes the role of a Security Champion, they might find security tools difficult to use. There are still companies who are adopting DevSecOps and trying to bridge between the developer and security teams with a Security Champion. Therefore, the new role could have the lack of security knowledge and still need time to gain the knowledge that security teams already have [10].

Motivation and interest

Fast code delivery, deadlines and competition affects prioritization of security. Skipping security tasks that are viewed as unimportant or unnecessary, cutting corners, lacking security knowledge or the understanding of how the security tools work are some of the reasons why security is not used during software development [4, 6, 33]. Software developers experience pressure to finish before deadlines, which could lead to having a focus on meeting the minimum requirements instead of putting in time and effort to implement security in their work [24].

There can be a lack of motivation to use tools and perform the actions to secure software, and a lack of motivation to participate in security activities if it is not obligated [6]. Lack of motivation can happen if there is no standard use of security to practice, or if the customer is not asking for specific security measures to be taken. Developers already have pressure to create functional code and deliver them in time, so to have additional pressure of being potentially blamed for creating or not discovering security vulnerabilities could negatively affect their work. Having a blameless culture is therefore important to contribute to a security mindset [33]. This mindset will contribute to focus on how to learn from failure and figure out how to improve [5].

Individual interest for security seem to be a key factor for having security awareness and actions during software development [6, 33]. The people who have interest for security are usually people who engage in security discussion, both formal and informal, participate in security activities, and may have the role of a security champion in a software development team [33].

A need for a cultural shift

A motivation to change the company culture is to be able to reach the goal of getting high quality product faster into production. The silos of software developers and IT operations made it difficult to reach their goal as it would affect the optimization of the automation process and get standardized technology [15]. Removing the silos could pave the way for improved collaboration which will lead to the desired optimisation.

Unfortunately, as [15] discovered, anti-patterns such as snowball-er and headless chicken will stand in a way for the changes needed. The anti-patterns would be the result of companies not taking the cultural and organizational change into account or inquire the changes needed to be done beforehand. They might only use automation or cloud-solutions, or simply try them because they are trending and they want to follow the trends or the requests of clients who want to use it.

A serious challenge to use security in DevSecOps is that it cannot be

in the way of fast software development [5]. Another challenge is how using security tools in DevSecOps is not enough. The researchers of [24] observed that not implementing security from the beginning of a project where the teams do not know how to use DevSecOps methods, how the tools work and how to implement them will result into a software with security vulnerabilities. A foundation of security culture is needed where everyone follow the same security standard [33]. However, prioritizing security could be limited by time, budget, resource, and the need of improved tools [6].

When [24] followed a software development project assembled with several vendors, it was observed that the lack of culture resulted in a rivalry between the vendors where no one wanted to take the responsibility of the mistakes and rather blamed others for faults that had occurred in the project. The researches believed that if there was an established culture in the project with open communication, shared responsibility and empathy, then the rivalry may not have escalated to the level it did. While the goal during the software project was to transition to use DevSecOps, the lack of culture hindered it to do so.

2.2 Related works

In this section, studies that collected knowledge about agile methods, DevOps, DevSecOps, security and culture in the industry both in Norway and globally is presented.

In [6], semi-structured interviews with software developer consultants were conducted to see how they used software security individually and in teams, and how culture, customers, and material factors affected the use of software security. They discovered that a developer's individual interest in security played a huge role. Their interest could bring awareness to the rest of the teams through informal conversations and sharing knowledge. Some would also arrange activities for both team members and members outside the team to bring more focus on security, but it was expressed in their interviews how they noticed that the effects of the activities only lasted short-term. Customers could influence a developer team with their own wish to prioritise security, which were called high maturity level customers due to their experience with security and their work with sensitive and important data. Likewise, developers with an interest in security could influence a customer by bringing awareness to the customer about security. The developers had noticed that the customers have over the years had an increased interest in security, mostly with how it could negatively affect their company's reputation. A developer could have a security role in the team, but the authors state that the role is too vague and needs to be better defined as the role does not have specific security requirements to improve software development. It was suggested based on their results that the security role should be more defined, and all developers should have a

base-line course in security.

In [33], an empirical study was conducted focusing on DevOps' four pillars known as culture, automation, measurements and sharing (CAMS), and how companies integrated security into CAMS. DevOps was interpreted differently between companies. One interviewee believed that security is a part of DevOps, even though the word security itself is not in DevOps. For another interviewee, the term DevOps meant an increased focus on cooperation or automation, but not including security. If there is a security culture, it would be easier to spread the knowledge of security. However, there are split opinions of sharing knowledge. Usually, there are members of DevSecOps who specialize more in security and operate as a security team. Still, a security breach could still not be enough motivation in involving security in software development. One of the companies experienced a breach of security, but the management did not particularly react much, showing that an actual occurrence may not have an effect on using security.

2.2.1 DevOps and DevSecOps surveys

In [18], a survey was conducted by the researchers to investigate software development companies' DevOps culture in Jordan. They found the process of obtaining data slow as it was only a few years ago that companies in Jordan started to use DevOps methodology. In the survey, they asked about DevOps tools, experiences and face problems. According to their respondents, they spent several months to learn about DevOps principles. The majority also preferred practicing DevOps rather than practicing traditional methodologies. From reviewing the responses, the researchers concluded that it is possible to save costs if one tool was used for several of the DevOps phases. Regarding DevOps adoptions, the researcher discussed the employees' suggestions on how companies should learn that DevOps is a culture and start out with adopting microservices architecture and containerization. Companies should then practice continuous learning so they can spend time to learn and keep up with new tools.

In [10], a survey was sent out to those with either a background in developing or security to see what type of DevSecOps challenges they experienced. Their findings showed which experiences and opinions that were similar or different between the groups. Similar opinions and experiences shared were how security had improved after implementing DevSecOps, but their workload had also increased. They also shared similar views about automation tools to the CI/CD pipelines, not having enough time to take the opportunity for security training, and security tools could be difficult. While the roles and responsibilities were clear, the guidelines were not. The researcher also stated that the reason for similar answers from both groups came from working on the same issues with tight collaboration. Differences between the groups were their views

on the degree of how difficult security tools could be. The researcher believed that the knowledge gap between the developers and the security team is what creates this difference. An example is how to spot the false positives from the automated security testing tools. Security teams found security activities too laborious, and leadership found security testing tools too slow. The researcher pointed out that leadership and those with other security roles looked at the system as a whole, while developers and security team members look at one application at the time. Also, DevSecOps is still being implemented, so the tasks and knowledge are still new for those who are practicing and learning.

2.2.2 Observing dependencies

In [32], DevOps teams were observed in their meetings. The DevOps teams worked in larger projects, and had full responsibility for the features they developed, from the features' start of the development process to monitoring them. The teams had daily stand-up meetings, scrum of scrum meetings, spring planning meetings, and team leader meetings. Outside the meetings were ad hoc conversations, where teams would help each other without needing to schedule a meeting as they were not far apart thanks to the open work area. They also had boards for visually managing tasks, and used Slack and Skype for communicating digitally. Meetings were not only about giving status from the teams, but often about managing dependencies between the teams. The researchers identified the different dependencies in the large-scale projects, which were divided in type of knowledge-, process- and resources dependencies. They also identified what type of coordination mechanisms were used to manage the dependencies in large-scale projects.

2.2.3 The DevOps reports

The DevOps report for 2021 published by [21] investigated why companies considered being in the "mid-levels" of DevOps have not been able to reach the level of "high-levels" of DevOps. For some companies, it is unclear what DevOps actually is about and what the teams are supposed to do. While security awareness has increased, there are disagreements in whether to call DevOps for DevSecOps, as some believe that security in DevOps should be taken for granted. Others believe that it is important to include "Sec" in the name as it could normalise security in DevOps. Companies in mid-levels of DevOps have a culture that does not include all practices of DevOps, which is why they are still in mid-level and not in high-level. One of the big factors is to take the risk for change by encouraging risk-taking in their culture. Low-level companies discourage taking risks, which has the opposite effect because it creates increased risk, and slow and infrequent deployments.

The DevOps report for 2022 published by [25] explained how an organization's culture creates the foundation of how they use DevOps. If

they have a more generative culture, the chances are that their organization has higher levels of performance. They reported that there had been an increase of cloud-usage, either for public or private, which positively affected on the work culture. The same goes for an increase of using Site Reliability Engineering (SRE) as it improves reliability. Loosely-coupled architecture is described as the importance of how one component of the system can be created, tested and deployed without being dependent on another team or affecting other parts of the system. Cultural factors that positively impact the drive for security were factors such as cloud solutions, flexible work arrangements for employees, lower burnout, low turnover on teams, larger organization and feeling that the team is valued and invested by the business. Another factor was the investment on improving and maintaining high use of CI.

Chapter 3

Methodology

3.1 Overview

This chapter explains the chosen approach for this thesis, which is a mixed method of qualitative and quantitative research methodology. The qualitative method was used as the main method where semi-structured interviews were conducted. The quantitative method used surveys with follow-up questions as statements following the likert scale after the semi-structured interviews. The quantitative data was used as a supplement to the qualitative data. A selection of potential participants were contacted and given an information letter about participating in the study. Their personal data was anonymised and protected with the approval from Norwegian Centre for Research Data (NSD). The participants were divided into two groups, software developers and operations. As a privacy measure, half of the participants who did not have the role as a software developer were labeled operations due to their specified role.

Participants' answers were used as data to answer the research questions, presented in Section 1.2, of how they viewed and used DevOps, security, collaboration and their company culture. To cover all these topics, an interview guide was used during all the interviews, but given room for impromptu follow-up questions based on the answers and opinions given during the interview. After the interviews, interview transcripts were written and used for data analysis by performing thematic code analysis. A mixture of both inductive and deductive approach for the analysis was used to compare previous published literature of the topics while at the same time be open to new information and knowledge that might add to the previous knowledge.

3.2 Mixed Method Approach

For this study, a mixed method of qualitative and quantitative approach was chosen to collect data for the research questions. The qualitative method used by conducting semi-structured interviews collected in-depth answers to answer the research questions. The quantitative method used

by sending a survey with follow-up questions was used as a a supplement for the results from the qualitative method. Both approaches have their strengths and limitations, and a mixture of both can give the possibility to strengthen each others weaknesses. Both data collection and analysis are done separately before being compared and results discussed [12].

Figure 3.1 shows an overview of the workflow used for the mixed method approach. *Literature background* and *Methods* went back and forth as the plan for the study developed. The stages between *Methods* and *Interviews* also went back and forth when adjustments for the interview guide were needed. As for *Surveys*, its design was finished after *Interviews* were done, and created through *Methods*.

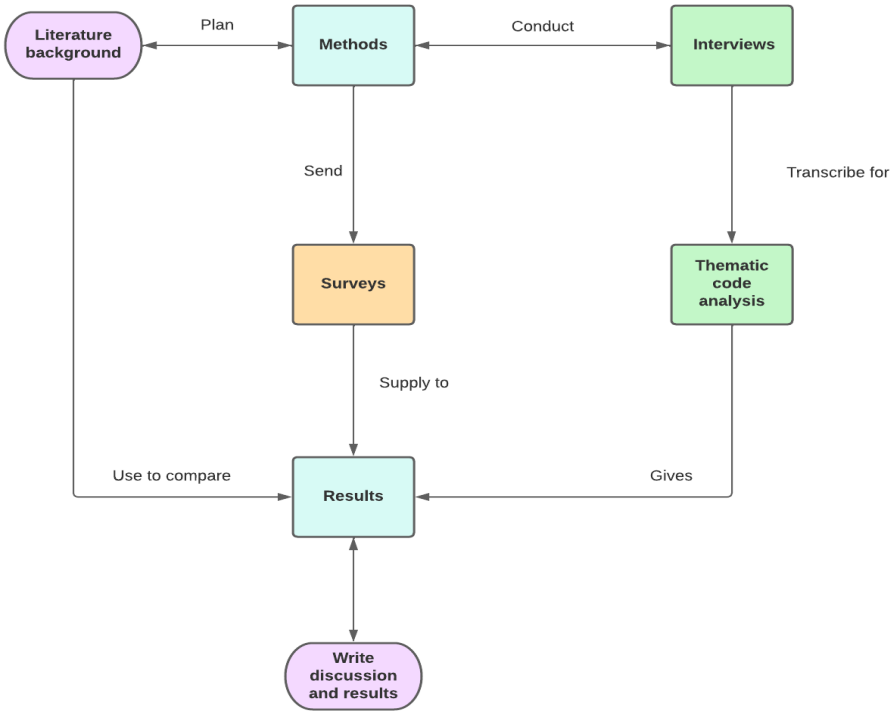

Figure 3.1: The workflow process of the approach.

3.2.1 Qualitative method with semi-structured interviews

A qualitative approach is suitable to gather data from participant's opinions, experiences, values, and perspectives explained openly in an interview [23]. Conducting semi-structured interviews for the qualitative study gives the flexibility to go in-depth to gain the object's knowledge [26]. Without this flexibility, valuable information may be missed if there is no in-depth discussion and follow-up questions outside the interview guide to find further answers. Considering the benefits of the qualitative approach and the qualitative data it can provide, it was the main method used for

this study.

Interview guide

An interview guide is used like a checklist to make sure the topics of the study are covered [26]. The interview guide was structured to address the research questions in order, starting with RQ1 and finishing with RQ4. RQ3 and RQ4 were combined in the interview guide. It was expected before the interviews started that the topics would intertwine together during the interviews, as they do influence each other. The interview guide can be found in Appendix A, translated from Norwegian to English.

The flexibility makes it possible to modify the interview guide when needed. It could be discovered from the interviews that some of the questions should be changed, added, or removed to collect better relevant data for upcoming interviews. Some instances, due to the natural flow of conversation, modifying the interview guide can also happen during the interview while staying on topic.

3.2.2 Quantitative method with surveys

While the qualitative method has a flexible design, the quantitative method used a fixed design with a set of questions so all participants answer the same questions in their survey. The benefits of a survey are how participants are anonymous, it is easy to send out to a large number of potential participants through email, and it be done quickly [26].

As the semi-structured interviews went with the flow of the conversation and the interview guide was edited along the interviews being conducted, it resulted to some of the questions not being asked to everyone. A survey would make sure that everyone would answer the exact same question, which further could be used for analysis.

Survey with follow-up questions

The survey was created with Nettskjema[1], as it provides a secure way to collect data through surveys and do not collect any personal data. A link was sent to the same participants from the interviews to answer. The survey was divided in three parts: DevOps and the development process, security and collaboration. The survey also included a page with questions from the DORA DevOps Quick Check[2]. This page was included if the participant practiced DevOps, but not made obligatory as it was possible the participant felt they did not practice DevOps and did not know how to answer. The reason why to use the questions from DORA DevOps Quick

[1]Nettskjema: https://nettskjema.no/

[2]The DORA DevOps Quick Check: https://www.devops-research.com/quickcheck.html

Check was because it was used by [25] to check the level of DevOps used by companies, which this thesis investigated too.

The survey used a likert scale, from "strongly agree" to "strongly disagree". Some questions would be sufficient with a "yes", "no" or "do not know" answer options. However, to make it easier to do the analysis of the answers, all questions excluding the last page of questions followed the likert scale. The survey can be found in Appendix B, shown as screenshots taken from Nettskjema.

3.3 Gathering previous works

Literature review is essential to find the recent findings in relevant research for the study [26]. By reviewing articles about the topics of DevOps or DevSecOps, security, collaboration and culture, it is possible to draw the identified problem domains from their conclusions found in these topics. Interview questions based on these identified problems can be used to discover whether software developers and other roles experience the same or similar problems occurring in their company today.

The articles were found mainly by searching on Google Scholar, while some were forwarded from the supervisors of the project. For the background literature, search words such as and variations of "DevOps", "DevSecOps", "software development security", "software development culture", "DevSecOps technology", "DevOps collaboration", "DevOps culture" and "DevSecOps culture". Some were found through references used in articles, including references for research methods. Search words to find out more about research methods were "qualitative method" and "thematic code analysis". Literature by [12, 23, 26] provided lot of information on how to use research methods.

3.4 The participants

All participants work either by writing code or setting up environments for developing and testing code, or both. The majority of the participants work in consulting companies. Snowball sampling was used to contact potential participants. Snowball sampling is an approach where potential participants are found through one's own and others network [23]. To accommodate time and place for the participants, the interview took place either physically or digitally wherever and whenever the participants had time to do the interview.

The participants were first asked if they were interested in participating to an interview talking about their work. It was also explained that the questions asked would be about their own opinions, experiences and knowledge about DevOps or DevSecOps, security and collaboration in their company culture. An information letter was sent before the interview

so the participants could get familiar with the project and what their rights of participating were. A copy of the information letter was brought to the interviews for the participants to sign. For the digital interviews, the information letter was signed electronically or the participant gave their consent in the recording.

The majority of the participants had worked for three or less years in the industry. Two of the participants had between three and five years of experience, while another two participants had around ten or more years of experience in the industry. Two of the participants worked in in-house company. However, it was not focused about the differences between in-house and consultancy companies, as it was deemed to be too few participants from in-house to provide more information on the differences between the two.

In Chapter 4, the participants are divided into two groups. The first group is the developers, as some of the participants work either as front-end, back-end or full stack developer. Being a full stack developer includes both front-end and back-end developing. Software developers in this thesis means someone who writes application code. The other group will be referred to as operations. Operations can be a broad term, as the role could include handling hardware, configure systems, operate systems or roles who work closely with operations by aiding them in their work without developing application code but writing infrastructure code.

In this thesis, operations are considered those who have tasks which includes to help create and setup pipeline environment for developers to develop application code. While the participants of this group did not have operations as a part of their title and do not view themselves as what is commonly known as IT operations, labeling them as operations is a measurement to make them unidentifiable from using their own specific role in this thesis. By separating the participants into two groups, it makes it easier to distinguish between the groups when presenting the results in Section 4. While a couple in operations had previous experience as a software developer, it was focused on their recent working role to make sure they gave answers that correspond to recent times in the IT industry.

The definitions made by [11] are used to categorize the companies' size based on number of employees. Companies with more than 250 employees are considered large, while companies with less than 250 but more than 50 employees are considered medium-sized companies. Those with less than 50 employees are considered small companies, and those with less than 10 employees as micro.

3.5 Data management of collected data

Seeing as the study collected personal data such as voice recognition and work place, an application to collect the personal data for research was sent to Norwegian Centre for Research Data (NSD)[3] for approval. The data was categorized as yellow data[4]. Therefore, the the anonmyised transcripts were encrypted and stored in OneDrive through the OsloMet's student access by using Microsoft's two-factor authentication. Pseudonymisation was used for participants, and the list (also known as "koblingsnøkkel") with the informants' name and their pseudonyms were contained in an encrypted flash drive separated from the research data.

For audio recording, Nettskjema has their own Nettskjema-dictaphone app[5]. The Diktafon app records audio files which can be sent directly to Nettskjema's browser. The audio files cannot be heard in the app, only in the browser which also requires the same Microsoft's two-factor authentication access through OsloMet's student access.

The OpenAi transcription program Autotekst[6] was used to automatically transcript most of the audio recordings. According the Autotekst's web page, the service is running on University of Oslo (UiO) servers which do not let the data outside of UiO's infrastructure. Autotekst is approved to use for yellow data. According to the approval from NSD for this thesis, the data collected was labeled as yellow data. To use Autotekst, the OsloMet's student access through Microsoft's two-factor authentication was required. When a transcript of an audio recording was finished, the audio recording was deleted from the program. After a transcript was finished, it was quality-checked by listening to the audio recording and following the transcripts.

The names of the participants and their company were not included in the study as a means to protect their identity by anonymising their personal data, as explained in Section 3.8. Sound- and video recording were only used for transcription and deleted when the research project was concluded. Personal data was removed from the transcripts before continuing on to use the transcripts for data analysis.

For the analysis of the transcripts, Notion was used to organize the data from the transcripts, as well for other notes and data analysis. Notion[7] is program where users are able to take notes, manage and organize notes, and create a note-database. For creating visualisation, Lucidchart[8] and

[3]NSD: https://www.nsd.no/

[4]Klassifiering: https://ansatt.oslomet.no/klassifisering

[5]Nettskjema-dictaphone: https://www.uio.no/english/services/it/adm-services/nettskjema/help/nettskjema-dictaphone.html

[6]Autotekst: https://www.uio.no/tjenester/it/lyd-video/autotekst/

[7]Notion: https://www.notion.so

[8]Lucidchart: https://www.lucidchart.com/

Excel were used.

3.6 Data analysis

The data analysis was planned to start after the transcriptions of the interviews had been finished. However, thoughts of similarities and differences were written down while the interview process were on-going and developed before the data analysis. The analysis used both inductive and deductive approach. An inductive approach focuses on creating codes identified from the data that has been collected, while a deductive approach creates codes based on existing concepts and ideas [8, 26]. The inductive method was used to find and create codes from the data, while the deductive method was used with predefined grouping with the themes for each of the research questions. For example, for RQ1, a predefined theme would be DevOps before doing the thematic code analysis. Still, there was always a possibility for a new theme to be created during the analysis.

3.6.1 Thematic code analysis

Thematic code analysis use coding and comparison to create and identify patterns and themes in thematic networks from data [8, 26]. It is considered to be flexible for qualitative studies where changes may happen depending on the findings during the course of the study [26]. The codes may start out as few before evolving into several codes as the analysis of the transcriptions proceeds. Later, the codes are analysed and themes developed. The themes are grouped together before creating a thematic networks as a tool to discover patterns [9].

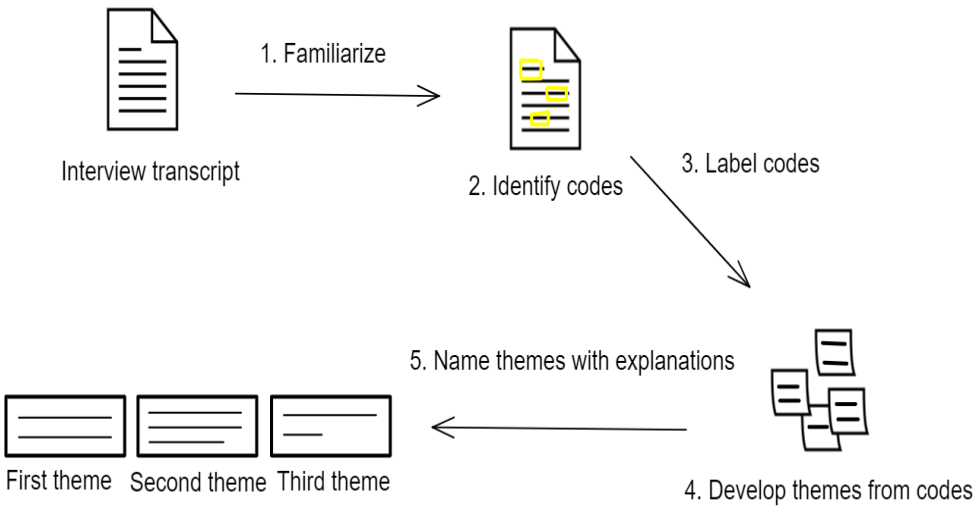

Figure 3.2: Illustration of the thematic code analysis phases suggested by [8].

As illustrated in Figure 3.2, the authors of [8] suggest to go through the thematic analysis in phases such as (1) familiarizing with the data, (2) identify code and label data that is considered relevant to the research questions, (3) find themes and their connections to each other, (4) evaluate themes, and (5) name and provide an explanation for the chosen themes.

The codes from the thematic code analysis can be seen in Table 3.1. The themes are in bold on top of the table, created from the analysis as one word representing each of the research questions with their related codes. It was evaluated to be easier to understand the one-worded themes as they are used to present the findings in Chapter 4.

DevOps	**Security**	**Collaboration**	**Culture**
Pipelines	Increased awareness	Slack	On-boarding
Automation	Responsibility	Pull requests	Documentation
Container technology	Customer requests	Meetings	Continuous learning
Cloud service	Security Champion	Roles	Open landscape
Continuous delivery	Trust in tools	Knowledge sharing	Trust in colleagues
Complexity	Knowledge	Code review	Overview of company

Table 3.1: An overview of the common codes found for each of the research questions from the thematic analysis.

3.7 Limitations

From the total of ten participants, five were software developers while the other five had other roles grouped together as operations in the industry. While the data had half of its answers from software developers and half from the operations group, the data was still limited from only ten workers in the IT industry. Including more participants would contribute to more data. However, as the study had a qualitative approach as its main method, the semi-structured interviews made it possible to have more time to gather in-depth data of insights of ten different workplaces in the IT industry.

For the interviews, some participants would answer vaguely to a question both from the interview guide and for a follow-up question. This made it hard to question further about a topic if the participant had no more to say. Vague answers would have to be evaluated whether it contributed to answer the research questions or not.

For the survey with the fixed questions, participants might find it difficult to answer some of the questions as they work with several projects with different type of working practices. As the likert scale was used to make it easier and more understandable of the analysis of the data, some participants might find it unfitting of how they wanted to answer.

3.8 Ethical considerations

When collecting any type of personal data, ethical considerations has to be taken into account with proper data management. How the data is managed is explained in Section 3.5.

According to the Norwegian Data Protection Authority[9], the basis of data management must be in place before the data collection can start. To collect personal data, an application was sent to NSD. Through this type of application, NSD guides research projects to cover all bases of data management. One of which was to have an information letter prepared for potential participants to read and sign. No personal data can be collect before the participant has given their consent to participate.

During the interviews for this study, a sound recorder or video recorder was used. Sound recordings reveal voice recognition and vocal expressions while video recording reveals a person's physical identity. Therefore, the data storage has to be secured and have authorized access for protection. That is why, as mentioned in Section 3.5, the files were encrypted before being stored in OneDrive through OsloMet's student access. OneDrive is owned by Microsoft, and as explained in Section 2.1.3 of the background chapter, the same GDPR laws do not apply to providers outside the EU/EEA. The encryption of the data before storing the data to OneDrive is a security measure with this issue in mind.

Other ethical considerations was to make participants feel safe. Before going to an interview, preparations were done by going through the steps that [26] presented of what to do and say to ensure the participant feels safe and secure to share their information. They suggested what to avoid in consideration of how questions are asked that may not be received well. Preparation included properly informing participants of what the research project was for, show respect during communication, and accept that they can withdraw their participation any time without being coerced to continue [26]. If someone were to withdraw from the project, any data that had been collected would be deleted and not used in the research.

[9]Datatilsynet: https://www.datatilsynet.no/rettigheter-og-plikter/virksomhetenes-plikter/behandlingsgrunnlag/

Chapter 4

Results

This chapter presents the results of the data collected from the qualitative method with the semi-structured interviews and from the quantitative method with the surveys. In this chapter, the participants in the study are referred as informants. The chapter presents their experiences, opinions and knowledge of their work by following the structure of the interview guide after DevOps, security, collaboration. The interview guide combined collaboration and culture together. In this chapter, collaboration and culture are presented separately.

For each topic presented, the results of the semi-structured interviews are presented first where some sections includes quotes from the informants. Right after, the results of the survey sent out to the informants with follow-up questions in form of statements are presented. The informants answered in which degree they agreed or disagreed with the statements using the likert scale. Firstly, the results of all the statements are presented with their values collected in a table. Secondly, a figure with a bar chart diagram visually shows the results to better see which part of the likert scale holds the majority and minority of the values. Because of this, "Strongly agree" and "Agree" are combined, as well as "Strongly disagree" and "Disagree". What was considered the interesting values of the survey are presented.

4.1 The informants

Table 4.1 shows an overview of the informants and their companies that participated in the study. Majority of the companies were established more than 20 years ago.Regarding the size of the company, it was evenly distributed between small (3), medium (3) and big companies (4). The majority of the informants have less than 3 years experience in the industry, while two have less than 6 years experience, and two has more than 10 years experience. The informants were divided into two groups of developers and operations, as explained in Section 3.4.

Table 4.1 shows the companies named and organised from A to J. The

informants from each of the company is represented in the table with their years of experience and the type of role they have, either as a developer or operations. One informant represents one company. To refer to which informant talked about which company, informant A talks about their work in company A, while informant B talks about company B, and so on.

Company	Size	Age (years)	Experience (years)	Role
A	Small	>20	<6	Developer
B	Small	>20	<6	Developer
C	Medium	>5	<3	Developer
D	Medium	>20	<3	Developer
E	Medium	>20	<3	Developer
F	Small	>5	>10	Operations
G	Big	>10	<3	Operations
H	Big	>20	<3	Operations
I	Big	>20	<3	Operations
J	Big	>20	>10	Operations

Table 4.1: An overview of the companies and the informants participating in the study. The columns "Size" and "Age" refer to the company size and company age. "Experience" and "Role" refer to the informants' experience and their role.

As seen in Table 4.1, those with a developer role are from company A-E, while those with an operations role are from company F-J. Within each of the groups, the table is organised after size, then age. After their age they are organised after the informants' years of experience. None of the developers are a part of a big company, while none of operations are part of a medium company. Some of the informants have experience from either a smaller or bigger company, but they focused on talking about their current company.

The two informants with more than ten years experience in the industry shared their own experiences from their company, but also a lot of their own impression and experiences on what they had seen or heard from around the industry through the years. The rest of the informants shared mostly from their own experiences in their own company, as they had not been working for long in the industry.

4.2 DevOps

The first topic discussed about in the interviews were the informants definitions of DevOps, their impression of the term used by others in the industry, if they practiced it and how, and if they have experienced any type of challenges. While the term DevSecOps was asked about later in the interview, their answers are included here with the DevOps definitions.

4.2.1 DevOps and DevSecOps definitions

The definitions provided went from having no definition to elaborated definitions. Majority of the informants had heard about DevOps for the first time when taking their bachelor's degree, but did not what it was as the term was not defined but more talked about in passing. Some of them learned about the term when taking certificates involving DevOps or getting introduced to DevOps during their work in the IT field.

Almost all of the developers with less than three years experience were unsure of how to define DevOps. Majority of operations had more elaborated definitions compared to the majority of the developers. They emphasised on automation, the CI/CD pipeline, tighter collaboration and continuous learning and improvement. They explained that DevOps is a way of thinking and a way of working. Without this mindset to continuously improving and learning to make the process more efficiently, then a high degree of DevOps is not practiced.

> "DevOps is a way of thinking, a way of working. You do not sit there and do manual processes. It is a continuous way to work for development" - Informant G

When asked about the term DevSecOps, both groups easily understood that it involved security in DevOps. However, the developers who were already unsure of how to define DevOps had not thought about the extent to have security in DevOps. Those who knew of DevOps knew that it included security in all the phases of the CI/CD pipeline. Some of the latter mentioned that security is a part of increasing quality to a product, which is why it should be considered a natural part of the whole process.

Informant F expressed how it was unnecessary to have the term DevSecOps, as it should be a given to integrate security in DevOps as it is part of increasing product quality. This statement was agreed by those familiar with DevOps, while those unsure of DevOps understood why there was such a sentiment about DevSecOps. The same informant talked about the term NoOps. If developers were able to do the operations teams' tasks, then in theory, there is no need for DevOps because the operations team are no longer needed.

4.2.2 The perception of DevOps in the industry

Those who were unsure of how to define DevOps still had the impression that DevOps was known and used by many in their company and the industry. They also knew that some had more knowledge and practicing it than others. Those who knew of DevOps, which were majority of operations and one developer, had the impressions that DevOps was a trending word, and called it being a hype and a buzzword.

Informant H had witnessed when DevOps was being explained to several people who asked about the term. It was clear that those who asked did not understand the explanation at first. Informant H explained if someone were to truly learn and understand DevOps better, you have to put your mind into to properly learn it. A quick explanation is not enough. They also said that it was better to practice and experience using DevOps to understand it rather than trying to understand through explanations from others.

The two informants with the most experience remembered when DevOps was introduced over 10 years ago. One of them said it was a lot of confusion in the beginning about what it was, while the other expressed that they thought the label over the years had been misinterpreted. It was explained how the increasing amount of servers took too long to configure, which created the need to automate the manual processes for the servers. The confusion of what DevOps was stemmed from a lack of proper definition, and the tools needed to automate the developing process and collaboration were not in place as they are today. The misinterpretation for many is how they believe DevOps are the tools used when developing, but does not think it is a is a way of thinking and working with the development process and others.

4.2.3 Practices

Common for everyone was using pipelines with automation. For developers, they used it when pushing code, while operations either or both wrote infrastructure code or were involved to plan how to build and what should be integrated in the pipeline. It was more common to use continuous delivery than continuous deployment, though those who created the pipeline infrastructure said that they would create continuous deployment if a customer requested it. This also included the developers where it depended on the project and customer if continuous deployment were to be used. For deploying code, few mentioned the tool Octopus Deploy.

Common practices by everyone were to push code and have it automatically tested before or after someone in their team approved their code in form of a pull request. For many, the code went through automatic tests again before production. As they commonly practiced continuous delivery, another approval was needed before their code was pushed into production.

Azure DevOps was used in both groups, but not by all. Majority used Kubernetes and said they were able to scale up and down as needed. The environments where they used Kubernetes was already in place and created by someone else in their or another team. One developer said their company did not use container technology as they had to first find out if it was beneficial for their projects to use.

Regarding automation, informant D said that as a developer, they want it as easy as possible. Informant G explained that they never experienced software developers who preferred to do anything manually rather than have their work automated. If something were not go wrong, they could simply roll back to a previous version.

> "It is more fun, you have less situations where you did something six months ago and an issue happens, and you try to remember what you did six months ago. The whole process happens faster. I consider that an absolute benefit" - Informant B

Three developers from two small and a medium company could run their code through testing before needing an approval or review from someone else. For the rest, it was common to need approval after pushing their code into the pipeline. Majority of the developers and two from operations in big companies who wrote infrastructure code were able to have their code in pre-production or skip this phase and go right to push to production. Skipping this phase depended on the project or code. An informant said if the code had such approval to be pushed from testing and into production, it would be fine to do so.

4.2.4 Challenges

Three informants in the operations group from big companies shared similar experiences. One challenge was when the applications were not compatible as they had different versions and were updated all the time. This made it challenging to make everything work together. Informant J said that since companies use tools in so many different ways they will experience different problems. There are not always solutions available for these problems, like tutorials on the web or instead unanswered posts asking for help. Informant H found it frustrating when this happened, explaining that if one of the applications they used needed a third party program which were not compatible with the rest of the applications, it would only add more problems to the already complex paradox.

> "There is always a place where things go wrong because something does not work in that damned paradox." - Informant H

Informant F explained how a lack of competence from leadership might result into increased risk of problems as leadership believe they do not need operations. They continued to explain how it is easy to lose control over the abstraction layers between infrastructure and application when complexity

increases, especially for software developers who do not know how the infrastructure works. In addition, the informant knew the leadership's focus was instead on the demands to deliver product as soon as possible than to improve the software development process.

Informant G firmly believed that there is no reason to make a system more complex than needed. They told that it can be challenging when a system gets more complex and there are more layers of abstraction to have control over. If someone lost control, many parts of the system could affect another needlessly, even if there was only a small code change.

A challenge of using a cloud provider is when they are down. Informant C, a developer from a medium-sized company experienced problems when Azure was down, halting everyone's work as they were not able to do anything beyond working locally without being able to collaborate with others. All they could do was to wait for the cloud provider to fix the problems outside the company's control.

To stay within budget could also make it challenging to practice DevOps. Good solutions can get expensive to purchase and operate. Informant G explained that choosing good solutions can be expensive in the long run without proper planning. If a costly and complex solution has been chosen, it can be expensive for the company to go back if the solution proved not to be beneficial after all. Not only in terms of costs, but also the amount of time people had to spend to learn the tools. With their explanation, the informant pointed out that proper planning and preparation is important for a company to save costs and time.

4.2.5 DevOps survey

Part 1 - DevOps, the software process	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
You feel that you use DevOps in a high degree	1	4	3	2	0
You feel ownership over the product you are working on	1	7	0	2	0
You are open to change the way you work if your company wishes to	4	6	0	0	0
You have access to the production environment	5	4	0	1	0
You are allowed to change pipelines	6	3	0	0	1
You have taken a course or a certification which included DevOps	2	1	2	4	1

Table 4.2: The table shows the collected answers for the DevOps and the software process of the survey.

From Figure 4.2, the five values from the survey statements can be seen with some values standing out more than others. The last two statements are the only ones where "Strongly disagree" is chosen. One informant strongly disagreed being allowed to change pipelines, while another strongly disagreed in having taken any course or certification which included DevOps. The majority feel they have ownership over the product they own, but not completely to the degree where they could

have chosen “Strongly agree”. The statement with the highest of value of “Strongly agree” is where informants are allowed to change pipelines.

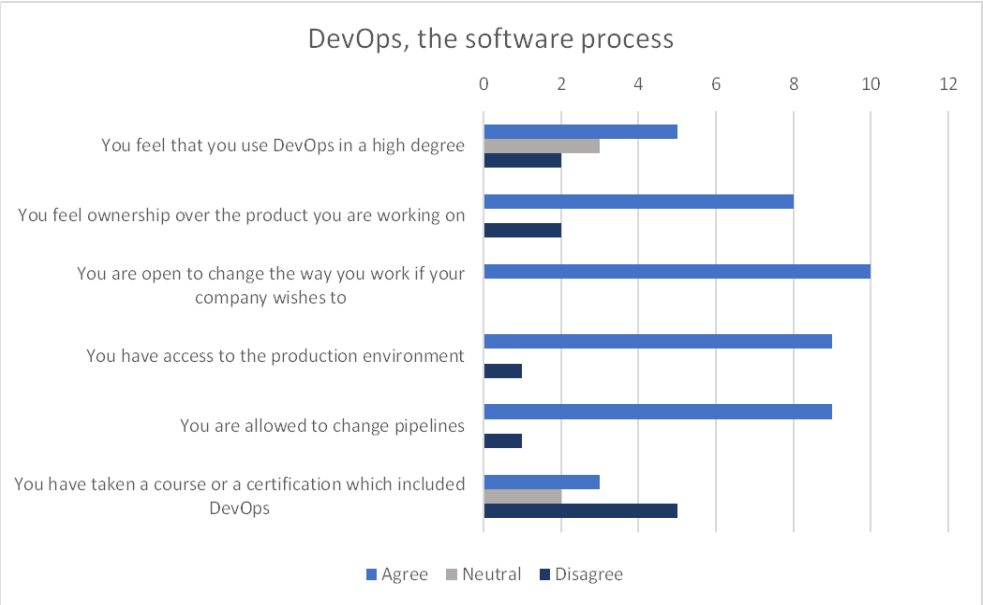

Figure 4.1: This bar chart diagram visualises the results of the DevOps and software development process part of the survey. See Figure 4.2 to see the values separately.

By looking at Figure 4.1, the bar chart diagram can give a picture of what the informants agreed, were neutral to or disagreed with the statements. The most dispersed statements are the degree of DevOps the informants feel they use and whether DevOps have been included in a course or certification they have taken. The majority feel they practice DevOps in a higher degree then the rest, while three informants feel they practice it neither high or low degree, and the minority of two disagree with the statement. These two statements are also the only ones with neutral values, where the the last statement of the two are the one with most disagreed values compared to the rest of the statements.

All informants are open to change the way they work if the company wishes to. The majority agree that they feel ownership of their product they work on. When it comes to accessing the production environment and being allowed to change pipelines, all but one informant in each statement agree that they are able to do so.

In Table 4.3, Table 4.4 and Table 4.5, the answers of the page where the survey included the questions taken from the DORA DevOps Quick Check are shown[1]. As this part of the survey was not obligatory to answer, some

[1]The DORA DevOps Quick Check: https://www.devops-research.com/quickcheck.html

of the questions were not answered by everyone. To see the full questions of this part of the survey, see Appendix B.

In Table 4.3, none of the informants have more than six months for lead time and time to restore. Only one informant has one to six month of lead time. The majority of the informants have one day to one week lead time and time to restore. Less than one hour lead and time to restore have only one value each.

	More than six months	One to six months	One week to one month	One day to one week	Less than one day	Less than one hour	Not answered
Lead time	0	1	2	4	1	1	1
Time to restore	0	0	2	3	2	1	2

Table 4.3: The table shows the answers for lead time and time to restore.

In Table 4.4, only one informant deploys on demand, which can be multiple deploys per day. None of the informants have a deploy frequency fewer than once per six months or between once per hour and once per day. Otherwise, the values are mostly distributed for deploy frequency between once per month and once every six months, between once per week and once per month, and between once per day and once per week.

	Fewer than once per six months	Between once per month and once every six months	Between once per week and once per month	Between once per day and once per week	Between once per hour and once per day	On demand (multiple deploys per day)	Not answered
Deploy frequency	0	3	3	2	0	1	1

Table 4.4: The table shows the answers for deploy frequency.

	0-15%	16-30%	31-45%	46-60%	61-75%	76-100%	Not answered
Change fail percentage	4	1	2	0	0	0	3

Table 4.5: The table shows the answers for change fail percentage.

In Table, 4.5, the majority has a change fail percentage of 0-15 percent. One informant has a change fail percentage of 16-30 percent, and two have 31-45 percent. None had a change fail percentage of 46-60 percent, 61-75 percent or 76-100 percent.

4.3 Security

For the second part of the interviews, the informants were asked about what they though about security, their own awareness and interest, if they were involved in security activities, used it for their work and if they had experienced any type of challenges.

4.3.1 Security awareness

All of the informants had notice an increase of security awareness in their companies, the industry and from customers over the years. Some mentioned how there had been an increase in media as well, as both attempts and successful attacks were reported publicly.

Informant E said that the media talked about the security risks of using the application TikTok. This brought up as a discussion in their company about whether or not TikTok should be on employees' phones. Because of this, the developer expected there to be updated security policies in the company.

Another developer said the Norwegian Data Protection Authority are being active in increasing security awareness. The developer told how security had never been excluded from their projects during the last years. If something was needed to be implemented to increase an application's security, the budget would make room for it.

The two informants with more than ten years experience explained that the reason why companies would strictly follow GDPR was because they did not want to get fined. One of them told about one solution to make it easier to follow GDPR was to choose an European cloud provider. If companies use a cloud provider outside of Europe, the companies had to take into consideration the fees and time for legal help to make sure they were within the rules of GDPR. Choosing an European cloud provider would save companies from this problem. However, the other informant said that companies prefer to use the biggest and more established cloud providers, even if they are outside of Europe.

4.3.2 Security interest

Majority of the developers were open to learn and use security if needed. Informant H told they had an interest in security, but were not as involved with security at the moment during the interview. They carried both hope and expectations to learn and use security in the future for their work, and were ready to take the opportunities when they were available.

Informant I told they had not started out being involved with security. After working on a project, they gained an interest for security and got

involved to learn and work with security. They found it difficult to understand the abbreviations when others talked about security. Eventually, they learned the abbreviations and thought they were simple to understand.

Informant F explained how software developers may not have the culture, understanding or motivation to think about integrated security and privacy. Informant A who said they did not have an interest of security, still felt that there should be basic security training offered and security policies followed by everyone. It was instead expected that the developers used common sense to make sure the code was secure.

4.3.3 Security involvement and practices

Majority of the developers who did not have a specific responsibility for security and had not worked for a long time in the industry were not particularly familiar with threat modelling, checklists, or what type of security tests were done. However, all informants were expected to at least use common sense, and follow any security routines and policies. The developers and a majority of operations were not familiar with static and dynamic scanning.

Majority of the developers knew there was accessibility restrictions set in place as a security measurement as it affected them when working. Informant C knew that the developers in their company had access to see users and the system. It was restricted to only certain developers with roles where they were allowed to read, change or delete parts of a code.

When receiving feedback from someone with a security role, informant D expected to be explained what is needed and how to do it. Code review was part of checking the security in each others code. At company I, there could be people who were negative to the feedback because it would increase their work, such as adding more code, mechanisms and authentication. Negativity would be met with the explanation of the strict security rules that need to be followed and there would be nothing further to discuss.

> "You just have to say that, for us it is very strict. If some of the security people has said that is the way it is suppose to be, then there will be no discussion about it." - Informant I

Company B, C and D had Security Champions. However, the developers from a medium- sized company said that they had not worked with them because they did not have the need to do so. They knew that they were able to go to the Security Champion if they had any security related they wanted to know about. They did not know what specific type of tasks the Security Champion did, but they did believe all security was handled by them. They assumed the Security Champion took care of everything involving security,

so the developers did not have to beyond the responsibility they already had. The developers of company D were already using third-party tools that were vetted, approved and used for a long time which they felt secure to use. Because of this, the developer trusted these tools as they put their trust in their Security Champion and others with the responsibility for new tools to make sure they were safe to use.

Informant I were involved with security activities. They had used threat modeling, and their company had their own intern vulnerability scanner. The informant found it easy to set up even if it took some time to understand how it worked. When asked about if developers should understand how vulnerability scanners work, informant I disagreed and said developers already have enough to do. They shared their experience when working with security architects. The security architects were only present in the beginning when working out the solution's needs. After the needs were settled and understood, the security architects were gone, the informant did not see them again during the project.

One of the developers had experience with working with and as a security architect, but had not been involved with threat modeling or using checklists. The security architect they had worked with had conducted a security risk analysis, but the developer had not been part of it. For security testing, the developer knew OWASP was used. An approval from someone outside of the team was needed if the team wanted to use a new tool.

Informant J believed static analysis of code is common in the industry, but companies may have the right tools but not using the tools so to fully cover the whole base. It was a positive opinion of informant J to find false positives after a security scanner analysis, because one could fix them before they later could turn into true positives. The informant knew that some companies would hire an attacker to check their security, and might not let their developers know beforehand. When the attack was completed, the hired attacker would present what they did, how they did it, and what the result was.

Company G run different tools to check against general security policies that they had integrated for their pipelines, and they run a strict zero trust policy, there are security in every step. The more barriers they set up, the harder it would be for an attacker to get through. Company I follows security policies which were created and recommended by outside trusted parties and the company's own security policies. Informant E knew that they used security packages to make sure older dependencies did not have security vulnerabilities.

4.3.4 Challenges

Some of the developers said they were up to learn more about security and use it if it was needed, but did express some concerns with increasing their

work. Informant D worried about how it would be too much extra work considering the developers already have a lot to do. They did believe that making it simple and easy as possible for the developers to use security tools would help. Informant I shared the opinion that developers already has too much to do and did not think it was a good idea to add more work for them. However, the informant still wanted all developers to at least know basic security knowledge.

> "We have developed it, we could have made it much more secure, but we do not have the time to." - Informant D

One of the big companies was one of many companies in Norway who experienced a security attack. One of their services suffered was harmed, and the company decided to not use the service's backup as it shared the same environment as its original. Their solution was to build the service from scratch. During the interview, the informant who told about the attack also brought up how National Security Authority (NSM) had been targeted as well.

Prioritising and improving security

The two informants with more than ten years experience explained that it is impossible to have complete security in all levels of a system as there always will be security issues somewhere. They added how adding tools to a growing system creates a more complex system with many layers of abstractions, making it challenging to keep up with security. If an attacker has put their minds to attack someone's system, with enough time they probably could. A strategy was doing a trade-off where it is prioritized to fix all the security issues on the attack surface rather than fixing the issues internally. Maxing out security defense on the surface increases the prevention of an attacker getting access internally where there could be many security vulnerabilities. In their experience, this strategy is good enough for many companies.

Informant J said is not only outside attackers to worry about when it comes to security. Network ports accidentally being open and exposed to the world is a security risks that has to be taken care of. The informant further explained the possibility of security vulnerabilities could result into accidentally leaking sensitive data to others without anyone having malicious intent.

4.3.5 Security survey

Table 4.6 shows the results of the security part of the survey. No informant chose "Strongly disagree" for any of the statements. While they are all in agreement that everyone should have some security responsibility, the

statement are the only one in the table where the values are split equally between “Strongly agree” and “Agree”. The first two statements which are more dispersed with the values than the rest are also the only ones with a single value chosen for “Strongly agree”.

Part 2 - Security	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
You know about OWASP Top 10 Web Application Security Risks	1	5	2	2	0
Your company offers a security course anyone can take voluntarily	1	4	2	3	0
You have an interest to learn about security	4	6	0	0	0
You want security in all the phases of your work	5	4	1	0	0
You believe that everyone should have some security responsibility instead of giving all of the security responsibilities to those with a security role	5	5	0	0	0
You want all new employees to take a basic course in security	6	4	0	0	0

Table 4.6: The table shows the collected answers for the security part of the survey.

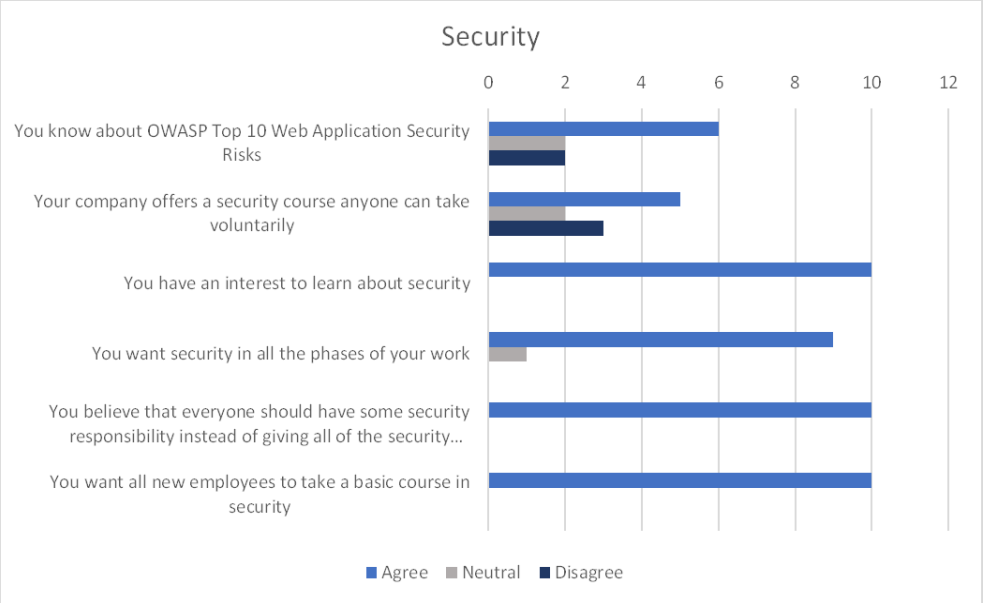

Figure 4.2: This bar chart diagram visualises the results of the security part of the survey. See Table 4.6 to see the five values separately.

In Figure 4.2, all the statements have the majority or all in agreement. According to their answers, everyone has an interest to learn about security, believe that everyone should have some security responsibility, and all new employees should take a basic course in security. Only one found themselves to be neutral when it comes to having security in all of the phases of their work, while the rest agreed. The first two statements are the only ones with all three values. Majority knew about OWASP Top 10 Web Application Security Risks, while two believed they are more neutrally familiar with it, while two others disagree in knowing much about it. Their companies are varying when it comes to offering voluntarily security

courses. For this statement, majority of the companies do offer this, while two chose neutral and three informants disagreed.

4.4 Collaboration

For the third part of the interviews, the informants were asked about how they collaborate and communicate together as teams, how they ask and get help, how they share their knowledge and what type of challenges they may have experienced.

4.4.1 The office

It was common to work in an open landscape office. Some had what they called hot desking or free seating, where employees could take any free desk they wanted in their office. If someone in the office wanted to work in a quite environment, they had to either find a quite room or use noise cancelling equipment. Company C and F had small soundproof booths for those who wanted to work quietly in the office.

For the majority, it was common to post questions on their teams communication channels or send a direct message to their colleagues. One from operations were conscious to not disturb others in their workflow, and preferred to send a message first. They would work with other tasks until they received an answer. However, if a task was urgent, they would most likely to go directly to their colleague at the office and ask for help.

One developer said they would get up and go directly to ask for help or information if it was possible, but would send a message if that was better. An informant from operations said it depended on how well they knew the person they wanted to ask for help from. If they knew them well, they would probably go directly to the person at the office. If they did not know the person well, they would send a message first.

One developer with less than six years experience explained they would evaluate which approach be most effective. If something simple would take more than two or three messages to send, then they would preferably go directly at the office as it was more effective and quicker. Another informant felt that messages could be perceived as annoying for some, so they preferred to be at the office to communicate and collaborate.

4.4.2 Ways to communicate work

There were common practices with everyone who participated in the study. Most use Slack to communicate with their teams or others. For the developers and operations who wrote software code or infrastructure, pull requests included an explanation of what the code was about. The code in the pull requests had to be reviewed before getting approved for further tests or merges.

> "We are good at communicating, we answer quickly, and we are humble so you can think one solution is better and someone else can think another solution is better. We also have good discussions about why. It is allowed to have different opinions." - Informant E

Small companies had it easier to communicate and getting an overview of what everyone was doing. One of them explained that start-ups have it easier to establish good communication, but they still need to adapt when the company grows. That is why having structure and balance bureaucracy is a must to make sure not to overwhelm processes and routines.

Informant D said it takes time build up the work relationships when there are new teams or roles created for DevOps. On the other hand, as explained by another informant, a start-up could practice DevOps immediately, especially if they have DevOps experienced employees and a smaller group of people to collaborate with.

4.4.3 Communication channels

One from operations said that developers preferred using Slack for collaborating. Aside from Slack, Teams were also used by many, but mostly for meetings and formal messaging. For requesting code reviews, asking for help, answering questions and form discussions about code work, Slack was found suitable. One developer favoured the program's availability to use emojis. Another explained how GitHub can be connected to Slack, making it easier to alert about peer requests and code reviews.

Several of the informants said their team had their own communication channel. Companies also had channels for subjects groups, collaboration between teams, departments and company news. Informant J said that channels could be created for an incident, and never used again. Some channels are so active, that some people leave them. The informant were surprised when people were able to keep up with the channels and their work. One developer said they did not keep up with the rest of the channels and only focused on the channels about their work.

> "I had seen a quote that many companies, they end up having more Slack channels than teams and people together. So yeah, usually there are channels. But again, it is always a fine balance about what is used and much is used." - Informant J

4.4.4 Sharing knowledge

If there was something new to learn, like a new tool, it was common for everyone to have one or few people to take upon themselves to learn about

it. Later, they would teach the rest of the team in form of a meeting or a presentation of what they had learned, problems encountered and how they had solved them. Both big, medium and small companies would send employees to conferences, workshop or courses to gain new knowledge and later share what they learned.

At company E, they were urged to post their question they had asked verbally in the office to the communication channel which others had access to. The reason was to include others to answer, discuss and learn. The same company shared documentation and news with all of the departments, and encourage each other to keep themselves updated and learn from each other. Informant E felt they were able find out what was going on with all the departments even if they were not participating in their work.

Company E, F, H and I had a weekly meeting that seemed to be more informal where they would share what they were doing, what they had learned or simply be social. These meetings were called breakfast meeting, brown-bag lunch meeting, or mini meetings.

4.4.5 Challenges

The informants shared some of the challenges they had experienced when collaborating with others.

Communication skills and understanding

A challenges experienced by informant I was how developers were not good at communicating their work, which made it difficult to assess what type of security was needed. The informant said that developers should take a communication course on how to communicate their work or learn how to visually present it so it is easily understandable for others.

A developer explained how the pull requests varied in their explanations of how to test the code. These explanations were needed as someone else was going to test the code before the code continued on in the pipeline. According to the developer, it was frequently told how people had to get better at writing about their pull requests.

Informant J, one of the more experienced informants explained a problem with collaboration between developers and operations was their different goals. Developers wanted to get their code quickly as possible out to production, while operations focused on stability rather than speed. For informant B, they felt the collaboration with the environment team improved when they learned what the team's needs were.

Another challenged informant J explained was how collaboration is depended on chemistry. If a team had no chemistry, then their collaboration

is doomed to fail. Informant B shared a similar opinion. Sometimes people do not match with their personalities, they way they work or with their opinions.

Who to ask in the team or company

Some of the informants in medium and big companies struggled to know which of the others have the right competence to give the help they needed, especially if the teams had increased in numbers. When several people had the same role in a bigger team, but not the same competence, it could take time to find out which one would give the right help. Another challenge could be that someone was very good at their role, but might not be good at working with people. On top of these challenges, those with the knowledge to help might not be available to as they were busy with their own tasks.

Company C solved this problem by using Slack channels. Their solution was creating a specific communication channel where people could ask for help. This made it possible for anyone to post a question, and whomever had the time and knowledge to answer could do so. This would save time looking for and go through several people just to find the right person with the right knowledge. The same company used to experience a hassle with how the communication channel would alert those that were not necessarily needed to be alerted. The company fixed this by creating teams in the channels, so if one question was to a specific team, the team could be addressed to with a tag "@", and everyone in that team would be alerted without unnecessarily alerting others. Through this, informant C said that not only will the right team receive a question or message they can answer to, but others part of the channel get the opportunity to learn from the question, discussions, answers, and know what others are working on.

One developer wished it was clear who the product owner was. If they had questions about the product they were working on, they wanted to have a person they could contact who were well acquainted of the product. One in operations shared a similar wish, saying if there was not an established product owner, then there would not be anyone who would voluntarily take upon the extra responsibility as one.

No meetings or too long meetings

One informant said that they experienced a period of almost no meetings, which made them feel they did not have an overview of what was going on in their department. The informant said that it seemed to be coincidental as people simply were unavailable to hold the meetings. They hoped that when the meetings would get back on track, they would be updated on current happenings within and outside the team.

Informant C and I shared their experiences of meetings they did not find useful. Informant C said that stand-ups and sync meetings, which are

suppose to be short just to give everyone an overview of their work status, could end up being too long. Sync-meetings for bigger teams could end up being prolonged as some went into detail of their work, which was not useful for informant C. Informant I said they would attend a meeting in case others attending needed the informant there. However, for some meeting the informant would realise during the meeting that they were not needed and do not need to know the work discussed.

4.4.6 Collaboration survey

The results of the collaboration part of the survey is shown in Figure 4.7. It shows that "Agree" has evenly values of 4 and 5 for all statements. Having a low threshold to ask for help has the highest value for "Strongly agree", while having an overview of all departments in the company is the only one where an informant strongly disagreed. Knowing who has what type of responsibility in other teams has no one who strongly agreed.

Part 3 - Collaboration	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
You have a good collaboration with your team	5	4	1	0	0
You have a good collaboration with those outside your team	3	5	1	1	0
There is a low threshold to ask for help in your company	6	4	0	0	0
You have a good overview of what is going on in all of the departments in your company	1	4	3	1	1
You know who has what type of responsibility in your team	2	5	2	1	0
You know who has what type of responsibility in other teams	0	5	2	3	0
You work closely with the environment team/developers	2	5	3	0	0
You work closely with those with a security role or security responsibilities	1	4	4	1	0

Table 4.7: The table shows the collected answers for the collaboration part of the survey.

Figure 4.3 visualise the results for the collaboration. The majority of all the statements are in agreement, but the statement where the informants feel there is a low threshold to ask for help in their company is the only one where everyone is in full agreement. One in each statement feel like they are neutral when having a good collaboration with their team and working closely with either the environment team or the developers, while no one disagreed with these statements.

When having a good collaborating with those outside their team, majority except for two agreed. One felt neutral and one disagreed. Half felt they had a good overview of what is going on in the other departments in their company. The minority disagreed, while three felt neutral. The same goes for those who know who has what type of responsibility in their team, and those who work closely with those with a security role or security responsibilities. However, as seen in the figure, the latter only differs with one between those who agreed and those who feel neutral. The only statement where those who disagreed surpassed those who are neutral is

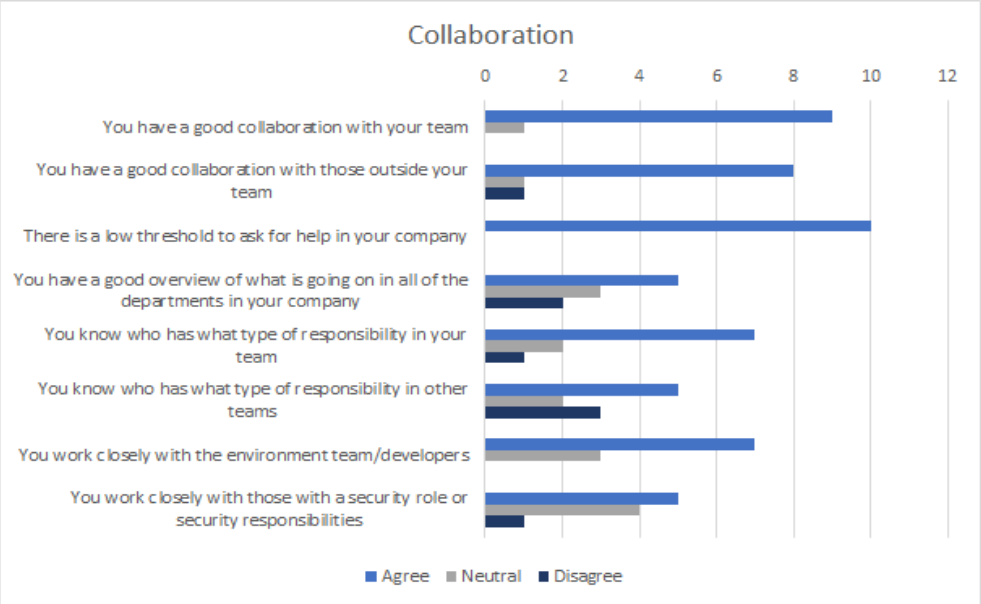

Figure 4.3: The bar chart diagram visualises the results of the collaboration part of the survey. See Table 4.7 to see the fives values separately.

the one where they know who has what type of responsibility in other teams.

4.5 Culture

During the third part of the interviews, asking about collaboration also included questions about culture. Collaboration and culture intertwine together as collaboration is based on culture regarding how everyone communicates and works together. However, as culture is not mainly collaboration, this section will present the findings of the informants answers how their company culture affects their work.

4.5.1 On-boarding

It was common that the companies had on-boarding for new employees. Others had internship, while some companies let the new employees work with their seniors to see how things were done. For the latter, company D hired people already with the experience and knowledge for what type of work their company does, and that they are capable to learn something new in a short time. Informant J said it is important with good chemistry in a team, which is why it is normal to hire new employees based on chemistry and how they can collaborate, or else it is doomed to fail.

Consultants go through a lot of on-boardings as they often work with new customers and projects. According to one informant, when starting at a new project the quality of the on-boarding depends on the customer's

culture. An experienced developer explained that a seasoned consultant would have experiences on-boardings and know how integrate themselves to a project.

Informant C had a great experience during their on-boarding process, but knew that others in the company did not share the same experience as the person who were suppose to aid them during their on-boarding process were too busy. One developer said they were continuously trying to make the on-boarding better at a project they were working on.

4.5.2 Overview of teams and company

Whilst being in the same team, an informant from the operations group had no need to know what the others in their team was doing as it was common to work on different projects. Still, they felt they had good knowledge of who to ask for help in the team if needed.

One developer said fluidity in teams could cause someone having a certain role even if others in the team have more experience for the role. This could make the distinction of who has what type of expertise in the team more difficult.

Informant E felt they had a good overview of the work in the company and its departments as it was part of their culture to daily post of their work which could be read by everyone in the company.

4.5.3 Knowledge sharing and knowledge transfer

Whenever someone was sent to a conference, workshop or course, it was common to share new knowledge in all companies. Several of the informants said that their team would learn a new tool for some time and share their discoveries, problems and solutions with the rest of the team.

Informant E was very satisfied of their knowledge-sharing culture across teams and department. What one department contributed with their knowledge, another could use in their work in their own way. An example given were if a front-end developer shared with the rest of the company what they were prioritising to create universal design, someone from the commercial department could use this information to advertise what they do for users to potential customers.

Two informants from operations explained they would do knowledge transfer when a new employee arrived or before an old employee left. Through either meetings or a a few hours sitting down, the person with the knowledge of an application would go through what they have done and how it worked so others could further develop the application.

4.5.4 Documentation

All informants document their work, but the degree and quality varied between people and budget. One developer explained that there is no common standard, but there are tools available to make documenting easier, such as Open API. Another tool used was an information security management system (ISMS).

According from both groups, documentation is dependent on budget and time. One developer said documentation is usually the first to go if there was a limiting budget. Even if people tried to create good quality documentation, budget may prevent them for doing so. Two from operations said it was up to the customer of how much could be documented, as they might want to prioritise get work done over documenting. The same was told by another developer where it was possible to search about questions in Slack to find posts with existing answers and discussion. This way, they could read previous Slack posts without asking themselves.

Informant E said their documentation was shared on a platform for everyone to use, either to share knowledge or learn from others across teams and departments. It was possible to search for specific documentation shared for anyone who was interested to learn in a topic.

4.5.5 Safety, trust and vulnerability

Developers who said they were happy with the collaboration and their culture emphasised the value of being social with your co-workers, feel safe with each other and trust each other so everyone could be vulnerable and feel safe to ask any questions. One developer felt that having a good relationship with you co-workers can make constructive criticism feel less personal. They trusted their co-workers enough to ask what the developer felt like were stupid questions. Informant D felt there were a good atmosphere and environment to work with their colleagues both inside and outside their team.

> "Yeah, we are all like... Bros." - Informant D

Informant B thought that IT consultancies in Norway were excellent at throwing social arrangements and promoting arenas and interests to join subject groups, recruiting and other social events. A social environment is important to create and build relationships and trust with your colleagues. Not only would it improve relationships, but it could improve and positively affect the company culture.

> "You should have a home even if you are hired out for a project in the consultancy branch." - Informant B

4.5.6 Challenges

While many was happy with their company culture, there were still challenges the informant or the company had experienced.

New employees and old employees

A challenge informants from both groups experienced when starting at their job was to figure out for themselves which person to ask for help and knowledge. Informant C would use the Slack function to see who was on the team. One informant said they did not understand anything during the discussions at the beginning as people used terms and abbreviations which were unfamiliar for someone new at the job. It could be confusing at first, and with no sort of list with explanations they had to spend some time trying to figure it out for themselves.

> "When you could see the team handles in Slack, you could click on it to see everyone who was on the team. But you did not necessarily know who were the experts." - Informant C

Three developers shared challenges with seniors. One developer experienced that employees who had worked for over a year or more were more sceptical to changes in practices of how they worked. This resulted to some not always fully following the changes which were tried to be implemented. Informant B said there could be a disagreement of how to work out a solution between someone with a lot of competence and someone with older competence. Informant E felt that seniors would fix a issue for them as it was faster. Still, the informant wanted to learn more of how the issues were fixed.

Writing and understanding documentation

The developer from company A said that there would be little motivation to document everything when companies change tools and structure every year. Another problem was how old and outdated documentation could be difficult to understand, especially if the person who wrote the documentation quit the company. They also said that documentation could be less prioritised because of deadlines and budgets.

> "It is better to read a little of what thoughts were had and how things are suppose to work than reading old legacy code written by someone from five years ago who does not work in the company anymore, and you have to decode abstractions in fourteen different levels." - Informant A.

A common goal and focus

One of the challenges mentioned in Section 4.4.5 was how software developers and operations have different goals. The ones familiar with DevOps said how silos have to be removed for there to be closer collaboration to practice DevOps. Continuing on, they emphasised to focus on continuous learning and improving which were needed of not only software developers and operations, but the whole company. Especially leadership and management have to share the same goal for DevOps to streamline products, and not only focusing on the tools. This seemed to be one of the biggest challenges when practicing DevOps.

> "Think about what it is that will gives us the most effective way to streamline products out to our users or customers, and take it from there. It is a good vision, but it is not easy to achieve." - Informant F

Chapter 5

Discussion

This chapter discusses the findings from the interviews together with the results of the survey presented in the previous chapter. The chapter interprets, compares, connects and discusses the findings for each research question.

5.1 RQ1: How is DevOps and DevSecOps defined and in what degree do they practice it?

How DevOps and DevSecOps are defined tells us whether there is a commonly shared definition of the term or various definitions. Practices and tools used also show what is commonly used by the industry when practicing DevOps.

5.1.1 Definitions of DevOps/DevSecOps

The various definitions of DevOps provided makes it clear that there is no common standard definition of the term DevOps. From the observations, this is due to some informants not being entirely sure what DevOps is. For most of the developers, it seemed to be they did not think much of the term's definition as they were not involved in a DevOps specific role. Rather, they followed the company's work practices, used the tools required and collaborated with whomever they needed help from without considering if this was labelled DevOps. Those who were able to provide elaborated and similar definitions were involved with roles and tasks such as planning and creating pipelines with necessary tools, policies, and security which the software developers used for writing application code.

As pointed out by some of the informants in Section 4.2.2, there have been misinterpretations and mislabelling of the term DevOps over the years. Because of this, there is not one standard definition used by articles and blog posts as people have their own interpretations of the term. The DevOps report by [21] stated that some companies do not know what DevOps is, while [33] discovered from their research that companies interpreted DevOps differently. The DevOps term is still not fully clear for

everyone. It would be difficult to learn about a specific term when working in the industry when the term is not standardised for everyone.

While the focus on breaking down the silos and automate tasks were major factors to follow the CI/CD pipeline, cloud services and container technology are increasingly being implemented to the pipeline. The popularity of the technology has caused DevOps to be referred as a trend and a buzzword without it being commonly known what it is actually about, which [15] called a DevOps anti pattern titled "headless chicken". The impact of the popularity might have shifted the focus from what DevOps is to only about the tools used in DevOps.

Considering many informants had heard about the term but still did not know what it was in their bachelor's studies and most did not have DevOps in any courses or certifications, it is suggested that DevOps should be included in the higher education's curriculum. In this way, new employees in the IT industry will already been introduced to DevOps. According to informant H, one explanation about DevOps is not enough. In [18], many of the respondents spent several months to learn about DevOps principles. Therefore, learning about DevOps principles in an educational setting could provide useful instead of relying on few explanations and spending time trying to catch up when working in the IT-industry.

In [21], there was a disagreement whether to call it DevSecOps as security should be a part of DevOps without needing to expand the label. Some of the informants shared the view of not calling it DevSecOps. The other view of the term in the report states that DevOps should be called DevSecOps as a start to show people that security is indeed a part of DevOps from the beginning. Considering how there are people who do not know or misrepresent the DevOps term, it might be more beneficial to have an explicit DevOps definition leaving no doubt that security is included from the beginning and throughout the CI/CD pipeline.

5.1.2 Degree of practicing DevOps

During the interviews, it was clear that many of the companies in a transition to use DevOps. As presented in Section 4.2.5, known practices of DevOps [21, 25] were observed and the results of the survey showed there were companies in both low-, mid- and high level of DevOps. Informant E explained how they felt they had taken it for granted how their company was using agile methodology, and how the silos were integrated. This could be a sign that key elements of DevOps are already practiced efficiently. Nevertheless, considering how there are different degrees of DevOps practiced in the industry, knowing what DevOps is and how it should be practiced could be considered important to make sure people are mindful to continuously improve their work process.

Developers prefer simplicity when using tools in pipelines, and are not afraid to take risks to make it. It can be interpreted that DevOps is capable to fulfill the demand for easy and efficient work processes. If the pipelines involve environments with automated building and testing with the use of container technology and maybe cloud services, then software developers could be more accepting to these technologies as they could make their work easier.

The findings suggest that it is common to use continuous delivery than deployment. This can be a part of the DevOps transition, as many still prefer to decide themselves when a code is pushed into production. The findings further explain that developers preferred simplicity, and were therefore open to use automation. Hence, continuous deployment might be preferred in the future as it contributes to faster deployment of the code into to the production environment.

A challenge that was described was how different versions of applications, platforms and tools used are often not compatible. This is interesting considering how there are continuous improvement in DevOps to get the product ready as fast as possible, which increases the need for quick updates and patches to the pipelines. When a new version is released and causes a compatibility issue, then companies have to fix it themselves if there are no tutorials or updated documents on how to fix them. This could result in companies running into the problem again when there are new releases. When the abstraction layers increases, so does the challenge of making everything between the layers compatible.

5.2 RQ2: To what extend is security important to their software development process?

To find out in what extent security is used and thought of, in terms of awareness, interest, practice and prioritisation, the informants were asked about their thoughts, opinions, knowledge and experience with security in their work.

5.2.1 Basic security knowledge and understanding

Based on the findings, it seems like the majority of developers did not think much about security because it was not within their area of responsibility. Because of this, attempting to gain knowledge of what security activities the informants were doing had its limitations. Developers usually did not know what type of security tools were used, and security activities were assumed to be carried out by others with security roles or a product owner type of role. The developers felt no need to worry as they trusted the security check and approvals of new tools by others. In the operations group, they were more aware of how security was prioritised

and implemented for infrastructure vulnerabilities, such as implementing security externally rather than internally.

All informants agreed that security is important, and the majority agreed that everyone should have some responsibility regarding security. However, for this research it is uncertain of what the bare minimum responsibility should entail. While many knew of OWASP Top 10, it is not guaranteed if they use the list often. OWASP Top 10 was also known by many in the study by [6]. The list is most likely the first to be used when learning about security. In [29], a framework was developed to practice cybersecurity in current agile activities, supporting continuous security in DevOps. Nonetheless, without basic security knowledge, software developers could face challenges being involved with continuous security without understanding how to practice it from the beginning. Informant I said everyone should at least know of the OWASP list and have a basic security course. There was a common consensus regarding this between the informants, as shown in Section 4.3.5. However, as also shown in the survey, not all companies offer a security course. Even if it is expected to use common sense to check if the code is secure, people need to learn what common sense of security actually is. People come from different educational backgrounds, have different experiences and knowledge. Therefore, assuming everyone has basic security knowledge is a security risk in itself. Therefore, it should be a standard for all IT companies to at least offer a basic security course for their employees.

5.2.2 Security awareness influenced by the media

While security awareness, interest and prioritisation has increased over the years, there was a surge of security awareness as warnings of expected increase of attack attempts appeared publicly in Norway due to the Ukraine-Russian war. Several companies and organizations experienced successful outside attacks or attempts to their systems.

There appears to be a consensus that security should be used and included in all phases of their work, even if they did not know how it specifically knew how it would be implemented. This could be expected because of the numerous warnings and strongly recommendations of improving and increasing security from the media. As this is publicly known, there are most likely few to oppose to the increase and improvement of security. However, deadlines affect the degree of prioritising security as shared by some of the informants. This was also observed by [24].

Another effect of media warnings could be how employees in the IT-industry understand that ignoring security could affect their career in the long run. All informants had an interest to learn about security, as shown in Section 4.3.5. The interest might not only come from a personal interest of security, but from a motivational one. They know security will always have an important role in their work to improve the quality of their product and

services. It could be said that the need to learn about new and improved technologies to keep up with competition and demands is the same for security. To not attain security knowledge may affect their work in the long run, which could be the cause of the increased interest.

It is almost completely impossible to have a 100 percent secure product. In Section4.3.4 in the previous chapter, the company who suffered an attack prioritised security and followed strict security policies and used their own security tools. One of the responsibilities of NSM is to warn about possible security threats in Norway. Still, they suffered an attack as well. Evidently, any company targeted in Norway is always vulnerable to attacks, making prioritising and improving security still crucial.

5.3 RQ3: How do they collaborate together and what impact do their collaboration have?

Collaboration is a key factor for to practice DevOps. The informants were asked how they collaborated with others, knew who to ask for help, what tools they used and how they approached others.

5.3.1 Digital or physical collaboration

It is common to use communication channels within teams, between teams, between departments or the whole company. Digital collaboration supports collaborating when working remotely, quick messaging, not physically disturbing others' workflow, connecting code work to communication channels and share documented work. Furthermore, it promotes having written discussions, including everyone part of the communication channels without being there in real time or in person.

The informants had an open office, either at their own company and usually at their customer's company. The informants who enjoyed being at the office shared the same sentiments as [32], who observed a company where they had an open office landscape. The open office created an environment made it easier to collaborate with their teams, across teams, and have ad hoc conversations.

Working at the office impacted the type of communication approaches the informants chose. Hot desking or free seating could be practical for collaboration as people who work together might simply choose free desks close to each other. Not only does this promote collaboration, but for big teams it could also provide the opportunity to sit next to others who they might not frequently talk to. An informant who preferred to be at the office could have non-work conversations. This promotes building up relationships with co-workers, making it easier to ask for help or discuss and share knowledge amongst each other which they might not would have done digitally.

There seems to be no standard way to approach someone, as it depends on knowledge, relationships, time and practicality. Having the knowledge to know who to ask depends on whether knowing about others' roles or knowing their expertise and competence beforehand. If there is a specific issue, it can take time to find out who to ask for help. For some informants, it took some time working at their company before figuring out who to ask for what type of issues. If there are good relationships, then most find it easier to reach out to someone. However, as one informant explained, they are mindful of not to disturb others workflow. Another shared how approaching physically could require less time for a quick chat than sending messages back and forth. Both are depended on the individual and the company culture, as each individual have their own preferences and experiences, and the company culture might influence on the type of approach people choose to collaborate.

5.3.2 Slack and peer requests

Slack is by far the most common communication program used in the IT-industry. It gives the flexibility to create communication channels without limits. Nonetheless, there is no guarantee that all channels will be active or if everyone added to the channel will engage with posts and comments. According to informant J, many channels are abandoned. Abandoning or ignoring channels seem to be based on the individual, as one informant used to ignore many of the company channels while another informant liked to get an overview of what was happening in the company. However, there seem to be no particular consequences if some channels are ignored, as people engage themselves in the important channels such as their team channel. For those who like to be updated and engage in multiple channels, Slack gives no limit for them to do so, giving them the opportunities they might not have outside the digital collaboration tool.

Pull requests can be seen as a way to require collaboration. As part of their work process, the informants pushing code were required to get approvals through code reviews. Pull requests could be commented and discussed, furthering collaboration to solve an issue or get clarity of what the code is for. Pull requests have to have sufficient description of what it is about and how to test it. If the pull requests lacked the information, then it would take more time to either figure out how to test the code or ask the person who sent the pull request to explain better.

5.3.3 Who to ask for help and where to share the answers

A common problem seem to be finding the right person to ask for help, especially in a bigger team or a medium to big company. Establishing defined roles and responsibilities would help knowing who to turn to for asking about help, clarification or acquiring knowledge. Time spent trying to solve this problem could instead be used for productive work.

A solution suggested was creating a specific channel to ask for help, and those with time could answer. Although this saves time to find the people who can help, it depends on the company culture and whether people are actively checking the channel. If the channel is inactive then the channel's purpose is gone. This is why encouraging posting questions and answers should be common practice in a company culture to keep the channel active.

5.3.4 To improve collaboration

Different roles can have different goals. This is why common goal setting and having an understanding about the other roles' responsibilities could help ensure that all roles work toward the same goals. Understanding each others' roles could increase collaboration as they know how to meet others' needs. This was experienced by informant B, who found it easier to collaborate with their environment team as their understood what the team wanted.

The majority of the statements in the survey have neutral respondents. Seeing as it was discussed in Section 5.1.2 of how several of the companies seem to be in the transition to adopt DevOps, it can be considered that there is room for improvement for some when it comes to DevOps collaboration. Still, depending on the company or the project, there may not be a must to have such knowledge and collaboration as the statements are about.

5.4 RQ4: How does the company's culture influence their software development process?

A company's culture is combined with how they work, how they think, use their tools, and collaborate to create their work environment. This includes not only those working in IT, but all departments across all levels in the company.

5.4.1 Transparency and openness to changes

While the majority of the informants did not have much industry experience, it was interesting to see their own mindset concerning the current IT industry. As they were new, they were more open to learn new tools and programming languages, and adapt to work practices and communication. In contrast, one developer shared how it could occur that some seniors were more close minded to change how they worked. Working longer in the industry, learning and getting used to routines and certain work practices could be hard to change as some prefer to do what they are used to. Some may show resistance, but if a company has set strict policies, employees will have to follow them. However, a good company culture where employees feel safe to share their view of possible changes

might contribute in compromises of how they and the company want to do things.

In small companies, two software developers had different views of having an overview of what was going on in their company. The one with an overview was practicing DevOps in a high degree while the other was not. According to the survey in Section 4.4.6 in the previous chapter, some of the informants disagreed with the statements about having a good overview of departments and the company, and knowing who has what type of responsibilities in other teams. Having an overview could also be based on an individual interest. If there is no reason to have an overview of what is going on in the company and rather be focused on ones own work, they might not care about what is going on outside of their own work. Still, there could be people who have a natural curiosity of what is going on without it being a obligatory knowledge to have. They may simply seek out information on their own to satisfy their curiosity or wanting to feel updated on company affairs. If a company does not have an internal platform or tools where its news and information are easily accessed, then they might have to actively ask around.

5.4.2 Integrating to the company

On-boarding impacts how new employees being integrated to the company's culture. For consultants, it affects how they are integrated at the start of a new project with a customer. There are no standard way to do on-boarding for all companies. The on-boarding process should be continuously optimised.

As a new employee, you will ask a lot of questions, even questions with answers which seniors considers as given and common knowledge. A safe environment contributes for new employees to know that it is okay to ask any type of questions and make mistakes without being blamed or yelled at. Making mistakes is a method of learning. Therefore, it is important to establish a blameless culture.

Social arrangements are common to create and build connection. Better relationships with co-workers creates an environment of feeling safe enough to ask "stupid" questions, voicing opinions and respect each others disagreements while being able to work out a solution. While building relationships should be a continuous effort, creating a safe environment for new employees should be one of the top prioritises.

Sync meetings seem to be crucial to update and let everyone know what is going on within the company and projects for their customers. In [32], they mapped out dependencies that helped syncing within the company. It seemed like that the informants who participated in both formal and informal meetings felt updated and in sync with their own and other teams.

Still, balancing the meetings considering time and information are needed as some might find the meetings unnecessary or time consuming.

5.4.3 Personality

Being at the office were a common practice before the pandemic. However, after the pandemic many preferred to keep working from home as they did under the pandemic. Some decided they would do both. As there are different type of personalities and method of working, those who prefer to work alone may choose to work remote. This however, can create a challenge when trying to reach those working remote or getting them to the office.

As informant J explained, there are people who are very good at their job, but not very good at working with people. As another informant said, if those who holds important knowledge are not available, then others will face challenges they otherwise would handle with the experts help. Combining isolation and bad communicating skills could affect collaboration. Finding capable team members could be a challenge [27]. Considering how DevOps is dependent on good collaboration, team members who are difficult to collaborate with could affect practicing DevOps.

5.4.4 Continuous learning

Knowledge sharing is crucial for continuous learning and improvement. People go to workshops and conferences, or take courses to gain new knowledge and share them with the rest of the team, department or company. Knowledge is also shared through documentations.

Documentation were prioritized to a various degree between the companies. Most would try to document what they were doing if they could, though some said pull requests would usually help show what the codes were for. Documentation does not have to be a document in a traditional sense. It can be in form of searchable posts in communication channels, with discussions and solutions.

Continuous learning were considered as part of the DevOps culture. From on-boarding and collaboration to improving the pipeline and acquire new tools and knowledge, to practice a high degree of DevOps would mean a high focus on continuous learning.

5.5 Limitations

When interviewing one person, this thesis only gets their perspective of their company culture. It is possible that someone else from the same company has different opinions and experiences of the company culture.

The study had only ten informants to participate. While a qualitative approach derives more information from in-depth conversations, ten informants are still a small group compared to the number of people working in the IT industry.

While this thesis covered DevOps, security, collaboration and culture, there could still be other factors that affect the software development process which were not included in the scope of this thesis.

5.6 Future work

Continuing on the findings of this research, other topics and questions can be researched to discover new data.

This research focused mainly on consultancies where employees are often working with different customers and projects outside their own company. Conducting a similar research only for the public sector could shed light on whether companies in that sector have similar or different type company culture. It could be interesting to see how companies in the public sector practice DevOps, security and collaboration.

Further research could be observing a company's culture, including management and leadership. What do each role prioritise during the software development, how do they prioritise security and how do they communicate when collaborating with others.

Continuous improvement of the CI/CD pipeline should be one of the main focuses when practicing DevOps. Researching and compare what type of tools are integrated in projects' pipelines for both quality checking and security testing code could provide findings of which pipelines are the most efficient of delivering code.

Chapter 6

Conclusion

This thesis researched the work process, knowledge, experiences and opinions from a group of ten people working in the IT field through semi-structured interviews and a follow-up survey. The interviews and survey were divided in three parts: DevOps, security and collaboration. These parts were topics researched through literature and former similar studies done before creating and structuring the interview guide. The findings of the data are the answers and discussions shared by the informants, which are presented in Chapter 4. The discussion of the findings for each topic are in Chapter 5.

The first part of the main findings were how DevOps and DevSecOps were defined and in what degree the informants do practice DevOps. To summary the definitions provided: DevOps can be defined as a way of thinking and working to continuously improve and streamline products. Manual processes need to be automated as much as possible. Developers and operations need to have a close collaboration, meaning previous silos need to be removed. They should either work in the same team, or developers have to learn operations' tasks so they are able to work independently of their environments and tools used. To better the pipeline for streamlining products faster, experimenting and continuously learning how to continuously improve the process are needed. Cloud solutions and container technology can be used to improve the process if used correctly.

All were practicing DevOps in some degree, and it was clear which informants had a lot of experience of DevOps and which had not. However, there is no agreed standard definition of the term DevOps, which was revealed during the interviews. Various DevOps definitions were given, and few were not able to give one. Those who gave more elaborate definitions had experience with working with DevOps. Still, those who did not know how to define DevOps or was confused about the term were practicing DevOps to a degree at their workplace. The majority had only heard about the term through their bachelor's studies, some knowing that a voluntarily DevOps course was offered. However, as DevOps is currently being adopted by more companies, it is suggested

based on this research that a mandatory course should be provided so students who enter the IT industry are aware and knowledgeable of what DevOps is. From the findings, DevOps is instead known as a hyped trend, a buzzword thrown around in the industry without knowing that it is more than using tools. With an established standard definition of DevOps and how to practice it, the misinterpretations of DevOps could be reduced and properly understood and adopted.

The second part of the main findings were to what extent is security important to the informants' software development process. Everyone knew the importance of security, but not everyone was aware of how it was implemented. It was easily understood by most informants that DevSecOps meant including security in DevOps. As security should be an integral part of DevOps, the DevOps experienced informants believed the term DevSecOps is unnecessary. Implementing security in every phase of the CI/CD process seems not to be generally thought of by developers. From the findings, there are developers who do have a security interest and involve themselves with security activities and roles. However, the majority of the developers were not familiar with security activities and tools used. Based on their experiences, opinions and the increasing need of security, it is suggested that all IT companies offer a basic security course for all employees.

The third main findings were related to the informants collaboration and their company culture. Communication channels seem to be the key for most to establish collaboration between team members, multiple teams, projects and subject groups. Slack was well-known by everyone, and used to tailor groups and alerts for pull requests, discussion posts and support. Personality, relationship, digital and physical presence affected how collaboration were done. Some prefer to work in isolation, and some prefer being at the office. The latter typically being an open office which promotes a social environment for collaborating. Knowledge sharing and knowledge transfer are being done through formal and informal meetings, workshops, collaboration and documentation. There are various degrees of documentation quality as there are not set standard of writing documentation for everyone. Companies might set their own standards and use supporting tools to write, but time and budget could limit writing good documentation. Documentation can be shared between individuals, teams, departments and the whole company.

Finally, collecting and comparing all main findings give a whole picture of how companies' cultures influence their software development process. To start adopting using DevOps, a foundation of security practices, shared knowledge and a willingness to take risks must be in place. Rushing into using DevOps and new technology can result into too costly and complex solutions. Companies should take time to plan how they will adopt DevOps, which tools, resources and services matches their needs. It is suggested that DevOps is introduced during studies or new employees,

alongside with basic security knowledge training. A culture where employees feel safe to make mistakes, ask for help and discuss is essential. Social arrangements and environments support building good, trusted and vulnerable relationships with colleagues. This type of culture would help adopting DevOps and the encouragement to continuously learn and improve in an industry abundant with knowledge and new technology.

Bibliography

[1] Waheeda Syed Shameem Ahamed, Pavol Zavarsky and Bobby Swar. 'Security Audit of Docker Container Images in Cloud Architecture'. In: *2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC)*. IEEE. 2021, pp. 202–207.

[2] Fernando Almeida, Jorge Simões and Sérgio Lopes. 'Exploring the benefits of combining DevOps and agile'. In: *Future Internet* 14.2 (2022), p. 63.

[3] Ken Beck et al. *Principles behind the Agile Manifesto*. 2001. URL: https://agilemanifesto.org/principles.html.

[4] Karin Bernsmed and Martin Gilje Jaatun. 'Threat modelling and agile software development: Identified practice in four Norwegian organisations'. In: *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE. 2019, pp. 1–8.

[5] J. Bird. *DevOpsSec: Securing Software Through Continuous Delivery*. O'Reilly Media, Incorporated, 2016. ISBN: 9781491971413. URL: https://books.google.no/books?id=5v25AQAACAAJ.

[6] Nora Bodin and Hanna Kai Barstad Golberg. 'Software Security Culture in Development Teams: An Empirical Study'. MA thesis. NTNU, 2021.

[7] Kelly Brady et al. 'Docker container security in cloud computing'. In: *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE. 2020, pp. 0975–0980.

[8] Virginia Braun and Victoria Clarke. *Thematic analysis.* American Psychological Association, 2012.

[9] Virginia Braun and Victoria Clarke. 'Using thematic analysis in psychology'. In: *Qualitative research in psychology* 3.2 (2006), pp. 77–101.

[10] Camilla Colliander et al. 'Challenges of DevSecOps'. In: (2022).

[11] European Union Commission et al. 'Commission recommendation of 6 May 2003 concerning the definition of micro, small and medium-sized enterprises'. In: *official Journal of the European union* 46.L124 (2003), pp. 36–41.

[12] John W Creswell and J David Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2017.

[13] Datatilsynet. *Økt risiko for nettverksangrep*. 2022. URL: https://www.datatilsynet.no/aktuelt/aktuelle-nyheter-2022/okt-risiko-for-nettverksangrep/ (visited on 25/05/2022).

[14] Datatilsynet. *Utfyllende veiledning om Schrems II*. 2020. URL: https://www.datatilsynet.no/regelverk-og-verktoy/internasjonalt/retningslinjer-og-uttalelser-fra-personvernradet/utfyllende-veiledning-om-schrems-ii.

[15] Jessica Díaz et al. 'Why are many businesses instilling a DevOps culture into their organization?' In: *Empirical Software Engineering* 26 (2021), pp. 1–50.

[16] Digir. *Hva er Schrems II-dommen*. 2020. URL: https://www.digdir.no/handlingsplanen/hva-er-schrems-ii-dommen/2581.

[17] Docker. *Docker security*. 2022. URL: https://docs.docker.com/engine/security/ (visited on 08/06/2022).

[18] Ameena Flefil, Luay Alawneh and Firas Albalas. 'DevOps Culture in Software Development Companies in Jordan'. In: *2022 13th International Conference on Information and Communication Systems (ICICS)*. IEEE. 2022, pp. 167–173.

[19] Martin Fowler, Jim Highsmith et al. 'The agile manifesto'. In: *Software development* 9.8 (2001), pp. 28–35.

[20] Vipin Jain et al. 'Static Vulnerability Analysis of Docker Images'. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1131. 1. IOP Publishing. 2021, p. 012018.

[21] Nigel Kersten et al. *State of DevOps Report 2021*. 2021. URL: https://media.webteam.puppet.com/uploads/2021/07/Puppet-State-of-DevOps-Report-2021.pdf.

[22] Gene Kim et al. *The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations*. IT Revolution, 2021.

[23] Natasha Mack. 'Qualitative research methods: A data collector's field guide'. In: (2005).

[24] Jose Andre Morales et al. 'Security Impacts of Sub-Optimal DevSecOps Implementations in a Highly Regulated Environment'. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security*. ARES '20. Virtual Event, Ireland: Association for Computing Machinery, 2020. ISBN: 9781450388337. DOI: 10.1145/3407023.3409186. URL: https://doi.org/10.1145/3407023.3409186.

[25] Claire Peters et al. *2022 Accelerate State of DevOps Report*. 2022. URL: https://services.google.com/fh/files/misc/2022_state_of_devops_report.pdf.

[26] Colin Robson and Kieran McCartan. *Real world research*. Wiley Global Education, 2016.

[27] Morgan Rowse and Jason Cohen. 'A survey of DevOps in the south African software context'. In: (2021).

[28] SAFECode. *Software Security Takes a Champion: A Short Guide on Building and Sustaining a Successful Security Champions Program*. 2019.

[29] Hannes Salin and Martin Lundgren. 'Towards Agile Cybersecurity Risk Management for Autonomous Software Engineering Teams'. In: *Journal of Cybersecurity and Privacy* 2.2 (2022), pp. 276–291.

[30] Samar Al-Saqqa, Samer Sawalha and Hiba AbdelNabi. 'Agile Software Development: Methodologies and Trends.' In: *International Journal of Interactive Mobile Technologies* 14.11 (2020).

[31] Nasjonal sikkerhetsmyndighet. *Tiltak for å unngå tjenestenektangrep*. 2022. URL: https://nsm.no/aktuelt/tiltak-for-a-unnga-tjenestenektangrep (visited on 25/05/2022).

[32] Viktoria Stray, Nils Brede Moe and Andreas Aasheim. 'Dependency management in large-scale agile: a case study of DevOps teams'. In: *Proceeding of the 52nd Hawaii International Conference on System Sciences (HICSS 2019)*. University of Hawai'i. 2019.

[33] Nora Tomas, Jingyue Li and Huang Huang. 'An empirical study on culture, automation, measurement, and sharing of devsecops'. In: *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE. 2019, pp. 1–8.

[34] Ann Yi Wong et al. 'Threat Modeling and Security Analysis of Containers: A Survey'. In: *arXiv preprint arXiv:2111.11475* (2021).

Appendix A

Interview guide

Intro

1. Can you tell about the company you work for and what they do?
2. When was the company established?
3. How many employees are there?
4. What is your role?
5. What is your relevant background, like education and possibly previous work industry experience?

DevOps

1. How do you define DevOps?
2. How did you learn about DevOps?
3. What is your impression of what other know about DevOps?
4. How does the software development process take place? What are the steps from when you commit code to the different environments such as dev, testing and production?
5. Do you use cloud-solutions?
6. What tools and software do you use for the different steps of the software development process?
7. Do you practice DevOps at work by using Continuous Integration and Continuous Deployment/Delivery? Why/Why not?

Security

1. What do you know about the term DevSecOps?
2. When is security implemented during the software development process?

3. How do you assess the security needs?
4. Are you involved in implementing security?
5. What tools, software and guidelines are used to implement security and test the security level?
6. Do you and your team go through security training such as security courses, workshops or meetings?
7. Do you have a Security Champions? If yes, what are they roles?
8. Do you work with a security team? How do you feel about working with them? How do you respond to their feedback?
9. Do customers ask for certain security actions? Do you influence customers to prioritize security?

Collaboration

1. How do you collaborate with your team?
2. How do you collaborate with others outside your team?
3. When a project is on-going, is it clear for everyone who has responsibility of the different parts/stages of the software development process?
4. Is there something you wish were done differently when collaborating?
5. How do you document the work you do? How do you share it?
6. How do you share knowledge?
7. What type of arrangements do you have when there are new employees?

Closing

Is there anything else you want to share based on what we have talked about so far?

Appendix B

The survey

Del 1 - DevOps, utviklingsprosessen

Du føler du jobber i stor grad med DevOps *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du føler eierskap over produktet du jobber med *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du er åpen for å endre måten du jobber hvis bedriften ønsker det *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du har tilgang til produksjonsmiljø *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du har mulighet til å redigere pipeline *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du har tatt kurs eller sertifisering som inkluderte DevOps *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Del 2 - Sikkerhet

Du kjenner til OWASP Top 10 Web Application Security Risks *

- ◯ Svært enig
- ◯ Enig
- ◯ Ingen formening
- ◯ Uenig
- ◯ Svært uenig

Bedriften tilbyr sikkerhetskurs man kan ta frivillig *

- ◯ Svært enig
- ◯ Enig
- ◯ Ingen formening
- ◯ Uenig
- ◯ Svært uenig

Du er interessert i å lære om sikkerhet *

- ◯ Svært enig
- ◯ Enig
- ◯ Ingen formening
- ◯ Uenig
- ◯ Svært uenig

Du vil ha sikkerhet i alle faser av arbeidet ditt *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du mener at alle skal ha noe ansvar for sikkerhet, i stedet for å gi all sikkerhetsansvar til de med sikkerhetsrolle *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du mener at nyansatte skal ha et grunnleggende kurs i sikkerhet *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Del 3 - Samarbeid

Du opplever et godt samarbeid med teamet ditt *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du opplever et godt samarbeid med de utenfor teamet ditt *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Det er lav terskel i bedriften for å spørre om hjelp *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du mener at du har oversikt over hva som skjer i bedriften hos alle avdelingene *

Er du nøytral på skalaen kan du velge "Ingen formening"-alternativet

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du vet tydelig hvem som har hvilke type ansvar i teamet *

Er du nøytral kan du velge "Ingen formening"-alternativet

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Det er tydelig hvem som har hvilke type ansvar i andre teams *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du samarbeider tett med miljøteam/utviklere *

Er du utvikler, svar for samarbeid med miljøteam. Er du ikke utvikler, svar for samarbeid med utviklere.

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Du samarbeider tett med de med sikkerhetsrolle/ansvar *

- ○ Svært enig
- ○ Enig
- ○ Ingen formening
- ○ Uenig
- ○ Svært uenig

Frivillig - DevOps

Spørsmålene fra denne siden er tatt fra DORAs DevOps Quick test som ble brukt til *Accelerate State of DevOps 2022*, som er 2022's DevOps rapport.

Svar gjerne på disse spørsmålene hvis du praktiserer DevOps.

Lead time: For the primary application or service you work on, what is your lead time for changes (that is, how long does it take to go from code committed to code successfully running in production)?

- ◯ More than six months
- ◯ One to six months
- ◯ One week to one month
- ◯ One day to one week
- ◯ Less than one day
- ◯ Less than one hour

Deploy frequency: For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?

- ◯ Fewer than once per six months
- ◯ Between once per month and once every six months
- ◯ Between once per week and once per month
- ◯ Between once per day and once per week
- ◯ Between once per hour and once per day
- ◯ On demand (multiple deploys per day)

Time to restore: For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (for example, unplanned outage, service impairment)?

- ○ More than six months
- ○ One to six months
- ○ One week to one month
- ○ One day to one week
- ○ Less than one day
- ○ Less than one hour

Change fail percentage: For the primary application or service you work on, what percentage of changes to production or releases to users result in degraded service (for example, lead to service impairment or service outage) and subsequently require remediation (for example, require a hotfix, rollback, fix forward, patch)?

- ○ 0-15%
- ○ 16-30%
- ○ 31-45%
- ○ 46-60%
- ○ 61-75%
- ○ 76-100%