UiA University of Agder

Master

Designing the CI/CD Security Maturity Model

A comprehensive approach for organizations to evaluate the security posture of their CI/CD pipelines

Nicolai Knudsen Bjørntvedt
Knut Sæther

SUPERVISORS
Marko Ilmari Niemimaa & Paolo Spagnoletti

University of Agder, 2024
Faculty of Social Sciences
Department of Information Systems

Acknowledgements

We would like to say a big thanks to Marko Ilmari Niemimaa and Paolo Spagnoletti, for their invaluable guidance and insights during our master's thesis project.

We also express our appreciation to all the industry experts and their organizations who generously participated in the interviews, contributing greatly to our research. Additionally, we are thankful to our family and friends for their support and understanding during this period of intense focus.

Finally, we acknowledge and appreciate each other's contributions throughout this journey, as we have collectively gained new insights and knowledge.

Kristiansand,
June 7th, 2024

Nicolai Knudsen Bjørntvedt

Knut Sæther

Abstract

With the rise of DevOps, the automation of software development processes like CI/CD, particularly in integration and operation has increased significantly. Organizations are leveraging these advancements to accelerate development and increase the frequency of deployments. However, despite the emphasis on speed, the integration of security practices within CI/CD pipelines often lags, posing significant risks as exemplified by high-profile breaches like SolarWinds. The existing literature and industry tools lack comprehensive frameworks for adequately assessing the security posture of CI/CD pipelines. Although some frameworks exist, they often fail to thoroughly address all necessary security aspects in depth.

To address this gap, we have developed a model based on design science principles that allows organizations to assess the security of their CI/CD pipelines. This is achieved through a structured spreadsheet that converts responses into numerical scores, facilitating a bottom-up assessment of security across various focus areas. This model enables the visualization of security levels through multiple layers of abstraction, each representing a different aspect of security.

Our research included three multivocal literature reviews (MLRs). The first MLR identified critical focus areas essential for a security-oriented maturity model. The second MLR mapped specific security practices to different maturity levels. The third MLR investigated whether any existing security-oriented maturity models could be adapted. Our model, the CI/CD Security Maturity Model (CICDSecMM), is grounded in these literature reviews and enriched by insights from numerous interviews and a case study. Through iterative cycles of building, evaluating, and refining the model based on interview feedback, we conducted a final case study to validate the model in a real-world setting.

Table of contents

List of figures

List of tables

1 Introduction

As organizations aim to deploy releases faster and more frequently by using automation, challenges may arise in concern of security. Security of software releases is often treated as a non-functional requirement, which is handled at a later stage of the software development life cycle. Thus, the automation of software development processes, particularly in integration and operation has increased significantly, due to the rise of DevOps and DevSecOps. Central to facilitating these advanced technological processes is Continuous Integration and Continuous Delivery/Deployment (CI/CD) (Rajapakse et al., 2022).

Continuous Integration (CI) is a development practice for software development where developers integrate and merge code frequently, often on a daily basis or multiple times per day (Rajapakse et al., 2022; Shahin et al., 2017). Continuous Deployment is a practice for automatically and continuously deploying the application to production or customer environments (Shahin et al., 2017). In combination, these practices include automated building and testing of software, followed by an automated push-based approach for deploying the software changes (Rajapakse et al., 2022; Shahin et al., 2017).

CI/CD pipelines are essential in DevOps, facilitating extensive automation across various stages of software development. This importance is highlighted in Kumar and Goyal's (2021) research, which comprehensively details the multiple components that make up DevOps frameworks, emphasizing the pivotal role of CI/CD pipelines.

As organizations increasingly adopt DevOps to accelerate and automate software releases, they often encounter security challenges. Security is typically managed later in the software development life cycle, complicating its integration while maintaining DevOps' agility (Rajapakse et al., 2022). In their pursuit of automation, organizations focus on the technological aspects of Continuous Integration/Continuous Deployment (CI/CD) pipelines. These pipelines facilitate rapid production through small, incremental changes, contrasting with traditional waterfall methods. Consequently, the quick transition from code development to production inherent in CI/CD also applies to potential vulnerabilities, such as the swift introduction of malicious code into repositories (Zampetti et al., 2021).

Despite DevOps' focus on rapid delivery through high-velocity operations, this approach often sidelines security, leading to potential vulnerabilities in accelerated development environments. Traditionally, security practices may not align well with these expedited processes. Zhou et al. (2023, p. 445) highlight that “DevOps practitioners degrade the priority of security since they regard security as the biggest hurdle to rapid application development considering traditional security methods do not fit the DevOps pipeline and are an inhibitor to DevOps agility”. Furthermore, the application security technologies have not undergone the same drastic improvements as development tools adapted to DevOps, meaning that they still target the traditional development cycle (Rajapakse et al., 2021).

One of the central goals of DevOps is the quick release of software, which has prompted the integration of automated security testing. However, significant human involvement remains essential. For example, while aspects of penetration testing can be automated, configuration,

analysis of results, and conducting the tests themselves still require human expertise (Rajapakse et al., 2022).

In response to these challenges, methodologies such as DevSecOps and shift-left security approaches have emerged. These approaches incorporate security as a fundamental, shared responsibility right from the beginning of the development cycle, thereby enhancing security across both software and infrastructure (Zhou et al., 2023).

Despite these initiatives, CI/CD pipelines have experienced serious security breaches in the past, as seen by the SolarWinds incident (Martínez & Durán, 2021). In the SolarWinds incident, attackers exploited the CI/CD pipeline to insert malicious code into software updates, demonstrating a critical vulnerability in the integration and deployment process (Bajpai & Lewis, 2022).

The security risks associated with automated CI/CD pipelines have often been overlooked and underestimated in both academic research and industry practices. This oversight underscores the critical need for a focused and comprehensive strategy to secure CI/CD pipelines, particularly in light of the rapid evolution of software development. Notably, approximately 88% of 1425 surveyed software development companies were in 2014 planning to implement DevOps practices within the next 5 years (Rafi et al., 2020). Furthermore, considering the severe implications of incidents like the SolarWinds attacks, there is a pressing need to enhance the security of CI/CD pipelines. This need becomes even more relevant as the reliance on CI/CD for achieving automation within DevOps continues to grow.

Furthermore, teams utilizing CI/CD pipelines, including their leaders, often lack adequate tools and frameworks to effectively manage and assess the security posture of these pipelines (Shahin et al., 2017). To our knowledge, there are no existing artifacts that allow organizations to thoroughly assess the security posture of their CI/CD pipelines. This represents a significant gap in the resources available to practitioners and industry experts, especially as the use of DevOps and CI/CD pipelines continues to grow.

Existing frameworks provide some level of security for CI/CD pipelines, but they often fall short of covering all necessary aspects comprehensively. Additionally, these resources are usually scattered and not ideally suited for assessing the overall CI/CD security posture. I.e. the OWASP DevSecOps Maturity Model (DSOMM) does address some relevant concepts within CI/CD, yet it is primarily focused on DevSecOps. This broader approach to software development encompasses more than just CI/CD. Moreover, the current literature on this subject is fragmented, lacking a unified perspective on the essential elements required to effectively evaluate the security of CI/CD pipelines. This gap, combined with the growing trend of organizations emphasizing DevOps and CI/CD, could leave organizations vulnerable in an environment where threat actors actively seek to exploit weaknesses.

1.1 Research questions

Constructing clear research questions represents a fundamental step in any research study because they indicate what the study is about and convey its essence (Thuan et al., 2019). We have based the construction of our research questions on the frameworks and methods suggested by Thuan et al. (2019), which utilize various forms and patterns of questions. Our research aims to address the following main research question:

RQ: How can a maturity model be designed to facilitate evaluation of the security posture of CI/CD pipelines.

To answer this question, we have identified two supporting research questions which we aim to explore:

SRQ1: What are the critical/essential focus areas that must be considered when designing a security-oriented maturity model for CI/CD pipelines?

SRQ2: How can the security practices be effectively mapped to different maturity levels to reflect incremental security improvements?

The supporting research questions served as the foundation for the knowledge base and are addressed through a multivocal literature review. In contrast, the main research question focuses on synthesizing this information and leads to the development of a security-oriented maturity model for CI/CD pipelines.

1.2 Research approach

For this research project, we have adopted a qualitative approach that prioritized interpretive research. This approach allowed us to explore phenomena through the meanings that individuals assign to them, as discussed by Myers and Avison (2002). Our objective was to design a maturity model, which initially required understanding the key areas for assessing security. Following this, we needed to identify the practices associated with each area and maturity level. To gather comprehensive data, we have focused on conducting semi-structured interviews and case study evaluation.

To address our research questions, we utilized an inductive analysis method to synthesize the data collected from semi-structured interviews. This inductive approach facilitated the design of the CI/CD Security Maturity Model (CICDSecMM). The model comprises nine focus areas, 31 sub-areas, and 173 practices, which are distributed across three maturity levels: basic, intermediate, and high.

1.3 Thesis structure

The structure of the thesis largely follows Gregor and Hevner's (2013) recommended structure for design science research studies. First, we present the knowledge base, focusing on CI/CD security. This section will outline the focus areas identified in the literature, provide a rationale for these areas, and review existing security-oriented maturity models to determine how they align with our research problem and the need to design a new maturity model from scratch.

Next, we will describe our research approach, with an emphasis on design science research. This will be followed by a methodological description of the multivocal literature review, consisting of three parts: one for focus areas, one for security practices, and one for existing security-oriented maturity models.

Finally, we will detail the development and iterations of the CI/CD Security Maturity Model, concluding with a discussion and summary of our findings.

2 Knowledge Base

This section outlines the results and findings derived from a multivocal literature review, which focuses on identifying focus areas extracted from both systematic and grey literature reviews. These findings form the foundational knowledge base for our master's thesis. Initially, we will introduce the various focus areas, provide a rationale for each, and then discuss the results of our literature review of pre-existing CI/CD security-oriented maturity models.

The research approach and methodology for the literature reviews is described in section 3.2 Knowledge base.

2.1 CI/CD Security

2.1.1 Secrets

Securing secrets is critical in CI/CD pipelines, as emphasized by a recent study by Pan et al. (2024). This research revealed that about 25% (80,000) of the CI/CD pipelines they examined transmitted at least one credential through the pipeline. Additionally, Chickowski (2023) highlighted that the most significant risks to exposure and integrity within CI/CD pipelines arise from the insecure management of secrets. This includes practices such as hardcoding credentials and failing to adequately protect credential storage in development environments. A proactive measure towards enhancing security is the principle of periodically rotating static credentials. This practice serves as a foundational step in securing access controls. Advancing beyond this, the adoption of temporary credentials represents a more sophisticated and dynamic strategy for managing the lifecycle of credentials, ensuring a tighter security posture. This approach is advocated by security standards such as those proposed by OWASP (Krivelevich & Gil, 2022).

2.1.2 Container security

Container security is of high importance in the realm of Continuous Integration/Continuous Deployment (CI/CD) due to the critical and foundational role containers play in the automation and optimization of DevOps practices. These containers are not just tools, they are the backbone of modern software development, enabling developers to package, distribute, and run applications in isolated environments. This isolation ensures consistency

across different development, testing, and production environments, making the automation processes more reliable and efficient. As such, the security of these containers directly influences the integrity and security of the entire CI/CD pipeline. Any vulnerability in a container can be exploited to compromise the automated workflows, potentially leading to significant disruptions in the development process and the deployment of insecure applications.

To ensure container security, it's essential to focus on multiple critical aspects that drives the overall integrity of the containerized environment. These include the security of container images, orchestration, maintenance, privileges, and origin verification. Each of these areas is a key pillar in strengthening container security within the CI/CD pipeline.

In terms of practical measures to secure these aspects, continuously scanning container images for vulnerabilities is a foundational step (Shevchuk et al., 2023). This is a critical practice for identifying and addressing potential security issues rapidly. Additionally, Shevchuk et al. (2023) emphasize the importance of regularly updating the containerization platform and the host operating system to protect against vulnerabilities. From an Identity and Access Management (IAM) perspective, running containers as a non-root user is an effective strategy for minimizing the risk of security incidents (Patra et al., 2022). Furthermore, a practice for organizations to use trusted images and registries, ensuring that only verified images are allowed to run in their environments. This approach helps mitigate the risk of deploying untrusted or malicious components, thus safeguarding the CI/CD pipeline (Souppaya et al., 2017).

In summary, container security is crucial for maintaining the integrity and efficiency of CI/CD pipelines. By addressing the security of container images, orchestration, maintenance, privileges, and origin verification, and by implementing recommended practices, organizations can significantly enhance their security posture, ensuring the safe and effective deployment of applications.

2.1.3 Security testing

Security testing within CI/CD pipelines is incredibly important, much like it is for most systems in general. It enables the identification of vulnerabilities and allows for their mitigation before they can be exploited by threat actors, thus preventing potential exposure and other errors. In an article by Vasile et al. (2019), they underscore the significance of incorporating security concepts such as security testing into CI environments. Doing so can significantly enhance defense against cyberattacks by quickly identifying and addressing vulnerabilities (Vasile et al., 2019).

Multiple areas and aspects of CI/CD are very valuable to be subjected to security testing. For instance, securing the source code involves a line-by-line search for common vulnerabilities (Vasile et al., 2019), which can be achieved through a Static Application Security Test (SAST). Furthermore, securing the Infrastructure as Code (IaC), which is a major component of CI/CD and involves managing the infrastructure, is crucial. It’s important to ensure

scanners are in place to secure the IaC, to detect and prevent misconfigurations and insecure instructions in the IaC-files (Center for Internet Security [CIS], 2022).

Third-party dependencies play a significant role in CI/CD pipelines and can serve as a gateway for malware into your pipelines. It's essential to validate third-party artifacts using a Software Composition Analysis (SCA) tool to detect whether any vulnerable open-source software is used in the final product (Cloud Native Computing Foundation [CNCF], 2021). Lastly, ensuring that the code is free of embedded secrets is also an important aspect, where scans are performed to check for this (Bajpai & Kannavara, 2023).

2.1.4 Artifacts security

Ensuring robust artifact security within the CI/CD pipeline is crucial for the smooth release of artifacts during continuous deployment. An artifact, in this context, is a product of the software development process, generated during the CI phase and subsequently deployed during the CD phase (Pan et al., 2024). Consequently, safeguarding artifact security directly correlates with the integrity of the system and/or service.

There are several key aspects to consider in ensuring artifact security. Firstly, integrity verification is essential for establishing trust and ensuring the consistency of artifacts between the build pipeline and the deployment phase (Bajpai & Lewis, 2022). This involves signing each artifact during the build process and verifying the signatures during deployment to confirm their authenticity and integrity (CIS, 2022).
Additionally, securely storing artifacts is vital to minimize the potential attack surface. One approach is to store pipeline output artifacts in a secured storage repository, safeguarding them against unauthorized access and tampering (CIS, 2022). This ensures that artifacts remain intact and unaltered throughout the deployment process, enhancing the overall security posture of the system.

2.1.5 Pipeline security

Ensuring the security of CI/CD pipelines is crucial, as these pipelines are central to the software development process. They automate the transformation of raw source code into deployable artifacts through various tasks, including code compilation, testing, and packaging. Given their automation of critical development and deployment steps, any security vulnerabilities in the pipelines can compromise not just the software artifacts but also the underlying infrastructure (CIS, 2022)..

One critical aspect of pipeline security is ensuring the integrity of pipelines for every run or build. This can be achieved by implementing various pipeline integrity validation measures. For instance, using a clean instance for each pipeline run can eliminate the risk of data integrity breaches and unavailability (CIS, 2022).

Maintaining the security of the flow and task orchestration within the pipeline is equally crucial. It involves safeguarding the integrity of tasks arranged within the flow control to prevent malicious alterations by contributors and threat actors. One effective measure to

ensure this integrity is by requiring pull requests to undergo review before merging (Scovetta, 2020).

2.1.6 Software supply chain and third-party risk

Third-party components are integral to code development, constituting a significant portion of code in applications or tools through the reuse of open-source frameworks and repositories, as evidenced by research showing a range of 85% to 97% (Martínez & Durán, 2021). In the context of DevOps and CI/CD, modern development heavily relies on these existing libraries, often without undergoing Static Application Security Testing (SAST) or Dynamic Application Security Testing (DAST). Consequently, CI/CD pipelines reliant on third-party dependencies can be vulnerable to code vulnerabilities (Bajpai & Lewis, 2022). For instance, downloading container images from untrusted sources and vendors can introduce security holes into containers (Patra et al., 2022).

To mitigate these risks, recommend implementing security checks at each step of the package import process, validating content trust through signature schemes for pulled libraries, maintaining an organization-wide catalog of trusted packages and sources, and controlling access to external package repositories.
Additionally, ensuring the integrity of every pipeline dependency before use is crucial. This involves validating that dependencies are trusted and free from tampering by comparing their checksum to that in a trusted source (CIS, 2022)

2.1.7 Skills and awareness

DevOps encourages developers to take on security responsibilities. However, the lack of security skills and knowledge among developers can hinder this objective, leading to human errors and subsequent issues (Rajapakse et al., 2021).

It's crucial for organizations to prioritize awareness and skills training. A study in 2021 highlighted that many developers lack the necessary skills to effectively use security tools and lack knowledge in this domain (Rajapakse et al., 2021). Continuity with the CI/CD principle underscores the importance of continuous learning and proper training to master software security principles and keep security knowledge up to date (Larios-Vargas et al., 2022).
Additionally, emphasizing the need for security-specific roles can help bridge the gap in security knowledge. Research suggests that having dedicated security roles prompts developers to consider the security implications of their technical decisions, thus promoting a proactive approach to security (Larios-Vargas et al., 2022).

2.1.8 Configuration management

In the domain of DevSecOps, effective configuration management is crucial. Developers may inadvertently introduce vulnerabilities by overlooking best practices when configuring software and underlying infrastructure. For instance, relying on default configurations of security tools could leave applications susceptible to security issues (Rajapakse et al., 2021).

Patching is another critical aspect of configuration management. A study in 2023 revealed that numerous CI/CD repositories still use outdated versions of scripts and tools with known vulnerabilities (Pan et al., 2024). However, these risks can be mitigated by ensuring that CI/CD tools are regularly updated, reducing the likelihood of exploitation by threat actors (NSA & CISA, 2023).

Additionally, insecure configurations and access control settings in containers can lead to vulnerable containers, posing risks to all stored source files (Rajapakse et al., 2021). Organizations can address this challenge by employing tools and processes that continuously assess and enforce configuration settings across the environment (Souppaya et al., 2022).

Moreover, configuration management is crucial within pipelines. A compromised or misconfigured Continuous Deployment Pipeline (CDP) may allow malicious or unwanted code to infiltrate production environments, posing significant risks (Rajapakse et al., 2021). To mitigate these risks, all modifications in pipelines should undergo review before acceptance, and pipeline configurations should utilize infrastructure as code to ensure repeatability and consistency in build environments (Bajpai & Lewis, 2022).

2.1.9 Identity and Access Management

Identity and Access management within the CI/CD pipeline is a critical measure for managing permissions regarding who can access specific tools and resources. This is crucial for preventing incidents like the SolarWinds breach, as it helps safeguard against unauthorized access and potential attacks (Sysdig, n.d.).

The principle of least privilege is an industry-standard best practice within the realm of managing access control and ensuring security. It limits the exposure of sensitive information and system functionalities to the minimum necessary for users to perform their duties. For instance, restricting access to the production environment only to a few trusted and qualified users.

Furthermore, regularly auditing administrative user accounts is a well-established principle. This ensures that users with administrative permissions are granted access for valid reasons, reinforcing the security of the system. Regular audits also help in identifying and revoking unnecessary permissions, thereby minimizing the risk of insider threats and unauthorized access (NSA & CISA, 2023).

2.1.10 Monitoring

Ensuring the integrity of CI/CD pipelines relies heavily on visibility. By establishing a robust feedback loop, teams gain real-time insights into pipeline operations, allowing them to quickly detect and address potential vulnerabilities (Chickowski, 2023). Failing to detect vulnerabilities early in the pipeline can result in significant security risks and operational disruptions.

Ensuring the build environment is adequately logged is crucial, given its central role in Continuous Integration and Continuous Deployment (CI/CD) pipelines. Proper logging facilitates the investigation of bugs or security incidents and simplifies the reproduction of the environment when necessary (CIS, 2022). Moreover, maintaining comprehensive logs aids in monitoring and optimizing the build process.

2.1.11 Rationale for Focus Areas

Focus area	**Rationale/Importance/Why/Contribution etc..**	**Literature reference:**
Secrets	Securing secrets is critical in CI/CD pipelines, as highlighted by recent research. These studies have shown that approximately 25% of the CI/CD pipelines examined transferred at least one credential through the pipeline. Other research has pointed out that the most significant risks to exposure and integrity within CI/CD pipelines stem from the insecure management of secrets. This often involves practices such as hardcoding credentials and not sufficiently protecting credential storage in development environments.	Brukman (2023) Chau et al. (2023) Chickowski (2023) Codefresh (n.d.) Dancuk (2021) Gu et al. (2023) Koishybayev et al. (2022) Krivelevich & Gil (2022) Maayan (n.d.) Moriconi et al. (2023) National Security Agency [NSA] & Cybersecurity and Infrastructure Security Agency [CISA] (2023) Pan et al. (2024) Rahman et al. (2021b) Sysdig (n.d) The Hacker News (2023) Zhou et al. (2023)
Container security	Ensuring security within containers is crucial in CI/CD because of their foundational role within DevOps automation. These containers directly influence the integrity and security of the entire CI/CD pipeline, and any exploited vulnerability can potentially cause significant disruptions in application development and deployment.	Brukman (2023) Kulanov & Stepanov (2023) Leppänen et al. (2022) Morgenstern (2023) Moriconi et al. (2023) Pan et al. (2024) Rajapakse et al. (2022) Sysdig (n.d.)
Security testing	Security testing within CI/CD pipelines is crucial as it facilitates the identification of vulnerabilities and enables timely mitigation, ideally before exploitation occurs. By integrating security concepts such as testing, organizations can significantly enhance their defence against cyberattacks.	Codefresh (n.d.) Krivelevich & Gil (2022) Morgenstern (2023) NSA & CISA (2023) Sysdig (n.d.) Zhou et al. (2023)
Artifact security	Ensuring strong artifact security within the CI/CD pipeline is essential for the seamless release of artifacts during continuous deployment. An artifact, in this context, is a product of the software development process, generated during the CI phase and deployed during the CD phase. Consequently,	Bajpai & Kannavara (2023) Brukman (2023) Chau et al. (2023) Chickowski (2023) Krivelevich & Gil (2022) Kulanov & Stepanov (2023) Moriconi et al. (2023)

	safeguarding artifact security directly correlates with the integrity of the system and/or service.	Pan et al. (2024) Pecka et al. (2022)
Pipeline security	Securing CI/CD pipelines is crucial, considering their central role in the software development cycle. These pipelines automate the conversion of raw source code into deployable artifacts via tasks like code compilation, testing, and packaging. Given their automation of vital development and deployment processes, any security flaws in the pipelines can jeopardize not only the software artifacts but also the infrastructure they rely on.	Bajpai & Kannavara (2023) Brukman (2023) Chau et al. (2023) Gu et al. (2023) Krivelevich & Gil (2022) Kulanov & Stepanov (2023) Moriconi et al. (2023) NSA & CISA (2023) Pan et al. (2024)
Software supply chain and third party risk	In DevOps and CI/CD, modern development heavily relies on these existing libraries, often without undergoing Static Application Security Testing (SATS) or Dynamic Application Security Testing (DAST). Consequently, CI/CD pipelines dependent on third-party dependencies can be vulnerable to code vulnerabilities. For example, downloading container images from untrusted sources and vendors can introduce security holes into containers.	Bajpai & Kannavara (2023) Brukman (2023) Chau et al. (2023) Chickowski (2023) Codefresh (n.d.) Gu et al. (2023) Koishybayev et al. (2022) Krivelevich & Gil (2022) Kulanov & Stepanov (2023) Moriconi et al. (2023) NSA & CISA (2023) Palo Alto Networks (n.d.) Pan et al. (2024) Rajapakse et al. (2022) Sysdig (n.d.) The Hacker News (2023)
Skills and awareness	As DevOps promotes developers to embrace security responsibilities. Nevertheless, the deficiency of security skills and knowledge among developers may impede this goal, resulting in human errors and subsequent issues.	Akbar et al. (2022) Chau et al. (2023) Krivelevich & Gil (2022) Pecka et al. (2022) Rafi et al. (2020) Rahman et al. (2021b) Rajapakse et al. (2021) Rajapakse et al. (2022) Shahin et al. (2017)
Configuration management	In the realm of DevOps, proficient configuration management holds great importance. Developers might unknowingly introduce vulnerabilities by neglecting best practices during software and infrastructure configuration. For instance, depending on default settings of security tools could expose applications to security risks.	Bajpai & Kannavara (2023) Brukman (2023) Chau et al. (2023) Codefresh (n.d.) Koishybayev et al. (2022) Krivelevich & Gil (2022) Kulanov & Stepanov (2023) NSA & CISA (2023) Palo Alto Networks (n.d.) Pan et al. (2024) Rajapakse et al. (2022) The Hacker News (2023)

Identity and Access management	Identity and Access Management (IAM) in the CI/CD pipeline is essential for controlling permissions and preventing unauthorized access to tools and resources. This is key to avoiding incidents like the SolarWinds breach by protecting against potential attacks.	Ahmadvand et al. (2018) Bajpai & Kannavara (2023) Brukman (2023) Chau et al. (2023) Chickowski (2023) Codefresh (n.d.) Gu et al. (2023) Koishybayev et al. (2022) Krivelevich & Gil (2022) Kulanov & Stepanov (2023) Leppänen et al. (2022) Moriconi et al. (2023) NSA & CISA (2023) Palo Alto Networks (n.d.) Pan et al. (2024) Rahman et al. (2019) Rajapakse et al. (2022) Sysdig (n.d.)
Monitoring	Making sure the CI/CD pipelines stay intact relies a lot on visibility. When teams set up a good feedback loop, they get instant insights into how the pipeline works, so they can quickly spot and fix any issues. Neglecting to identify vulnerabilities early in the pipeline can lead to substantial security risks and operational disruptions.	Brukman (2023) Chickowski (2023) Koishybayev et al. (2022) Krivelevich & Gil (2022)

Table 1: Rationale for Focus Areas

2.2 Existing maturity models

A maturity model serves as an effective instrument for assessing a company's current operational status, prioritizing enhancement strategies, and monitoring the advancement of their implementation. Consequently, maturity models are valuable in addressing and managing these aspects efficiently (de Bruin et al., 2005).

Furthermore, a maturity model functions as an enabler, offering specific problem-solving capabilities to address pre-existing issues. It represents an artifact that is based on the principles of design science, a domain where artifacts are carefully developed to act as essential components of systems. These systems are specifically constructed to tackle and resolve complex challenges that arise in the interaction between humans and machines, with the maturity model being a prime example of such an artifact (Hevner et al., 2004).
In the development of maturity models, which serve as enabling artifacts at the heart of design science, certain requirements must be adhered to (Becker et al., 2009). These requirements are grounded in the seven design guidelines outlined by Hevner et al. (2004). In accordance with these requirements, the first of the 8 requirements proposed by Becker et al. (2009) is "Comparison with existing maturity models". This stage involves providing a justification for the development of a new maturity model by evaluating how existing models align with the problem at hand. It emphasizes that the new models might offer enhancements or further developments over the existing ones (Becker et al., 2009).

In the comparison outlined below, two distinct categories of maturity models are discussed: white literature and grey literature. Maturity models classified under white literature are typically derived from academic research and are peer-reviewed, contributing substantively to the knowledge base. These models are often found in formally published papers. On the other hand, maturity models associated with grey literature are usually not subject to peer review and are commonly found in more informal sources. These include government documents, committee reports, standards, technical documentation, and fact sheets.

Title	Authors	Year	Outlet	Literature
Security Maturity Self-Assessment Framework for Software Development Lifecycle	Brasoveanu et al.	2022	International Conference on Availability, Reliability and Security	White literature
A Roadmap to Continuous Delivery Pipeline Maturity	Hornbeek & Jones	N/A	pages.awscloud.com (web)	Grey literature
The OWASP DevSecOps Maturity Model (DSOMM)	Pagel & Prasad	N/A	dsomm.owasp.org (web)	Grey literature

Table 2: Overview of security oriented maturity models

2.2.1 SMAF

Brasoveanu et al. (2022) highlight in their paper that many software vulnerabilities stem from inadequate focus on security during the development lifecycle. The study addresses this issue by proposing a Security Maturity Self-Assessment Framework. This framework aims to thoroughly examine and improve security measures throughout the software development process (Brasoveanu et al., 2022).

The Security Maturity Self-Assessment Framework (SMAF) effectively merges the strengths of the OWASP DevSecOps Maturity Model (DSOMM), OWASP Software Assurance Maturity Model (SAMM), and the Building Security In Maturity Model (BSIMM). It aims to bridge their gaps, enhancing software security maturity. SMAF evolved by evaluating DSOMM, pinpointing and integrating missing activities from BSIMM and SAMM, thereby refining the framework's comprehensiveness.

The SMAF model comprises six assessment areas: Governance, Architecture and Design, Code Development and Review, Build and Deploy, Verification and Validation, and Operations and Observability. Each area has its subsections for a detailed assessment, where responses are scored and aggregated. While the framework's areas like Build and Deploy, Code Development and Review, and Verification and Validation are highly applicable, others like Governance may not directly align with our area of research and the context of CI/CD. The model serves as a robust basis for evaluating the Software Development Life Cycle (SDLC), which aligns with its primary objective. However, it does not comprehensively

address the unique requirements of CI/CD security. While some principles are applicable, it overlooks key areas specific to CI/CD, such as container security. Additionally, the model incorporates elements like incident response within its focus areas, which, although relevant in broader contexts, lie outside our specific focus and lean more towards an operational perspective.

2.2.2 A Roadmap to Continuous Delivery Pipeline Maturity

The article from AWS provides a detailed guide on engineering practices for continuous delivery pipelines, aiming to simplify the software development toolchain within AWS. While it aims to maintain a degree of neutrality, the focus on AWS services might limit its general applicability. The content is structured around five maturity levels, evaluating the dimensions of People, Processes, and Technology (PPT) (Hornbeek & Jones, n.d.).

However, while the guide thoroughly addresses various aspects of Continuous Delivery (CD), it doesn't fully explore Continuous Integration (CI) or provide a structured method for quantifying maturity levels. Instead, it presents a matrix outlining areas and practices, which may not offer the most precise measurement of maturity. Additionally, the guide has a somewhat vendor-specific perspective, which could affect its universality.

2.2.3 OWASP DevSecOps Maturity model

The OWASP DevSecOps Maturity Model (DSOMM), created by OWASP, serves as a framework to illustrate and prioritize security measures within DevOps strategies. This model is categorized into five focus areas and eighteen sub-areas in total, encompassing dimensions such as Build and Deploy, Culture and Organization, Implementation, Information Gathering, and Test and Verification. Additionally, it outlines five maturity levels ranging from (1) Basic understanding of security practices to (5) Advanced adoption of security practices (Pagel & Prasad, n.d.).

Regarding our area of interest, there are certainly overlaps, and the model holds transferable value, particularly since CI/CD is integral to DevSecOps. Although it's not a direct match due to its broader scope, several of its dimensions, sub-dimensions, and practices were taken into account in the design process of our maturity model.

Requirement	SMAF	AWS Roadmap	OWASP DSOMM
Design Process	Identified and addressed gaps in the existing models DSOMM, SAMM and BSIMM. Conducted preliminary validation with industry professionals. Ensured adaptability and compliance with standards like ISO 27001	N/A	N/A
Content	Assessment of the(your) software security capabilities	Maturity assessment for Continuous Delivery Pipelines	Maturity mapping between Levels(practices) and dimensions.
Measurement area and Maturity levels	Five business functions serve as the areas for measurement, which are assessed across three distinct levels.	Three dimensions are evaluated across five maturity levels.	Five dimensions evaluated across five levels of maturity

Table 3: Comparison of existing maturity models

2.2.4 Reflection about Existing Security-Oriented Maturity Models

When comparing the three maturity models - SMAF, AWS Roadmap, and OWASP DSOMM, we find that they do not adequately align with our research problem.
The existing frameworks, while operational and generally applicable in broader contexts, fail to meet the criteria for a Security-oriented Maturity Model specifically tailored to CI/CD security. Although DSOMM and SMAF do partially address security of DevOps, they do not concentrate exclusively on CI/CD. This lack of focus reveals a significant gap in the market, characterized by an unmet demand for a model that addresses security concerns within CI/CD more precisely.

3 Research approach – Design science research

The structure of this chapter is organized according to the elements of design science research (Hevner et al., 2004). It starts with an overall description of the key aspects of design science research, and how this relates to our project. Subsequently, the methodology for the conducted literature reviews examining the knowledge base is described. Then, the environment which contributed to the design and evaluation of the artifact is presented, followed by a description of the project's design and evaluation approach.

3.1 Design science research

Within information systems research, IT artifacts intended to solve identified organizational problems can be created and evaluated with design science (Hevner et al., 2004). March and Smith (1995) describe four types of design artifacts produced by design science research in information systems: constructs, models, methods, and instantiations. The produced artifact from this project is a model, in the form of a maturity model. A maturity model "consists of a sequence of maturity levels for a class of objects. It represents an anticipated, desired, or

typical evolution path of these objects shaped as discrete stages. Typically, these objects are organizations or processes" (Becker et al., 2009, p. 213).

The overall research approach for this project was design science research, with utilization of research methods such as literature reviews, interviews, and case study. The research framework and design science research guidelines by Hevner et al. (2004) and the maturity model design requirements and procedure model by Becker et al. (2009) are among the most recognized papers within the areas of design science research in information systems and design of maturity models. Thus, their work has highly influenced how our project was conducted. We illustrate our project by filling Hevner et al.'s (2004) research framework with details from the context specific to our research project in Figure 1.

The information systems research framework by Hevner et al. (2004) is illustrated (Figure 1) with the research in the center, the environment to the left, and the knowledge base to the right. In behavioral science, the two phases of the research are called development and justification, while these phases are called building and evaluation for design science research. For our project, the artifact to build and evaluate has been the CI/CD Security Maturity Model. To achieve a relevant artifact, it must meet a business need(s), and ultimately provide utility to be applied in the appropriate environment.

The environment where the artifact is supposed to be applied to consists of people, organizations, and technology (Hevner et al., 2004). The environment in our project was considered to be comprised of organizations with software development teams. The characteristics of such organizations may vary extensively, in size, sector, criticality, and location. This also implies that the maturity most likely ranges from beginners to highly experienced and knowledgeable organizations. The types of roles that were considered as part of the target environment for our project were software developers, development platform engineers, IT- and security architects, and CISOs. All of these roles are natural to include in the work with CI/CD security, since it directly affects their work and their responsibilities. Both managerial and technical staff have an interest in how the security of their CI/CD infrastructure and processes is handled. The most relevant technologies for our project are CI/CD platforms, the infrastructure of the development environments, and the deployment environments.

To ensure rigor of the research, the knowledge base which consists of existing foundations and methodologies must be appropriately utilized when conducting the research (Hevner et al., 2004). Literature reviews were conducted to make use of the established knowledge within CI/CD security and the existing maturity models within similar topics. Our approach for the design of the model was also influenced by past research about, or involving, maturity model design. When new insights were gained during the process, the knowledge base of the project expanded, which was reflected by refinements on the artifact as part of the design process.

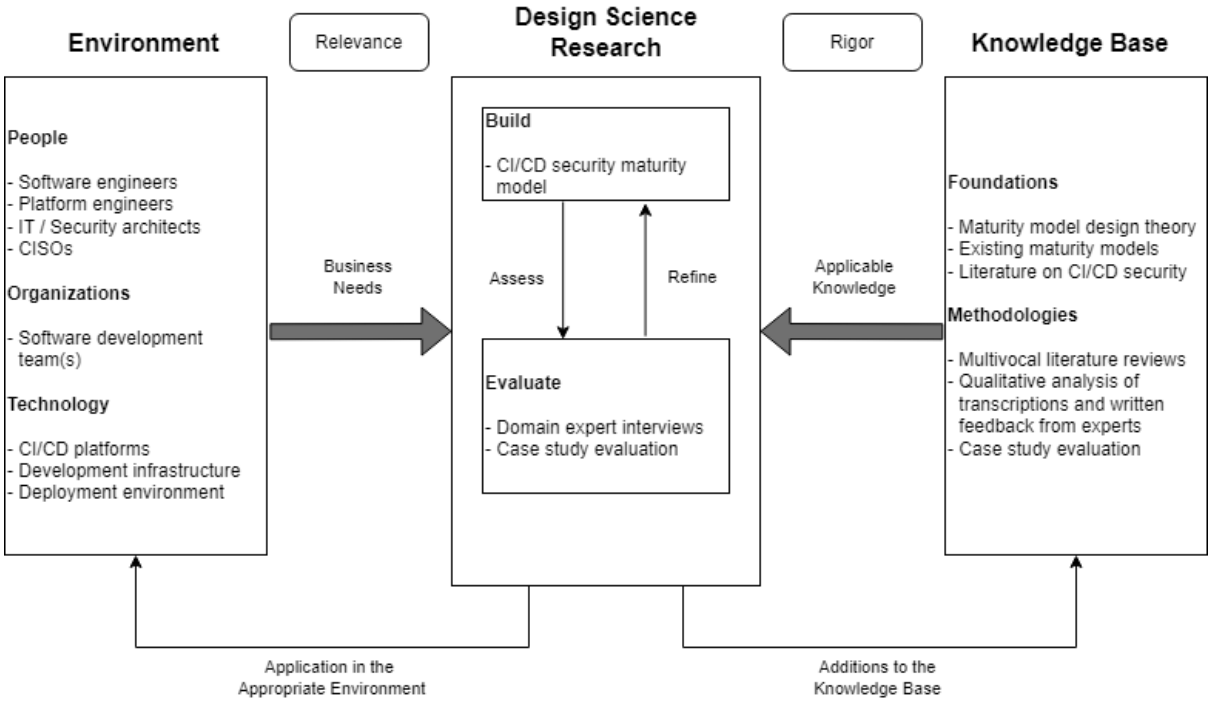

Figure 1: The CICDSecMM research framework (Adapted from Hevner et al. (2004))

3.2 Knowledge base

This section will present a rationale for conducting a multivocal literature review, which includes both systematic and grey literature reviews. Furthermore, it will present the three multivocal literature reviews performed, focusing on identifying the focus areas, security practices, and pre-existing security-oriented maturity models within the context of CI/CD pipelines.

3.2.1 Multivocal literature review

For our master project we have chosen to conduct Multivocal Literature Reviews (MLR) when it comes to performing the Literature Reviews (LR). This involves enhancing our research by delving into the realm of Grey literature (GL), besides the more conventional way of performing a LR which is to only utilize a Systematic Literature Review (SLR). While SLRs are crucial for both practitioners and researchers to pinpoint evidence and gaps in a particular research area, they tend to focus solely on formally published works, disregarding a significant body of "grey" literature (GL) (Garousi et al., 2019). MLRs, on the other hand, acknowledge the necessity of incorporating multiple perspectives, rather than relying solely on information rigorously reported in academic settings, i.e., formal literature (Garousi et al., 2019).

3.2.2 Systematic Literature review

We followed the 8-step process described by Xiao and Watson (2019), which is based on Kitchenham and Charters' (2007) guidelines (for systematic literature reviews in software

engineering) and Brereton et al.'s (2007) 10-stage systematic literature review process. Overall, Xiao and Watson's process is divided into three major stages: planning the review, conducting the review, and reporting the review. Even though the planning stage is placed as the first phase, it is not considered as a one-time event which is finalized before conducting the literature review. The process can be iterative, as problems or observations during the conduct of the literature review may result in the need for modifying the plans defined in the first stage.

In the first step, the problem is formulated through research questions which drive the entire literature review process (Xiao & Watson, 2019). The subsequent activities are supposed to be geared toward answering the defined research question(s). Thus, we started the planning-stage with formulating the research question. The next step involved developing the review protocol, which describes the elements of the review (e.g. purpose, research questions, inclusion criteria, and screening procedures).

When the review protocol was specified, the planning-stage was concluded and followed by the stage for conducting the review. The third step of the review process is to search the literature. We used all of the "three major sources to find literature" (Xiao & Watson, 2019, p. 103), electronic databases, backward searching, and forward searching. Our chosen electronic databases were Web of Science, IEEE Xplore, and Scopus. The keywords used for searching the literature were derived from the research questions and preliminary searches. We prioritized more exhaustive results over more precise results, to capture most of the potentially relevant literature and not miss some records. This prioritization led to the choice of using broader searches and keywords, which gave more irrelevant articles while reducing the risk of losing relevant articles. This adheres to Wanden-Berghe and Sanz-Valero's (2012) view on the balance between exhaustiveness and precision when scoping the bibliographic search. Only articles with titles that seemed relevant to our literature review were included for the next step.

The fourth step is screening for inclusion, in which we used the defined inclusion criteria from the planning-stage. For the screening, Xiao and Watson (2019) recommend following a procedure of two stages. This screening procedure starts "with a coarse sieve through the articles for inclusion based on the review of abstracts […], followed by a refined quality assessment based on a full-text review" (Xiao & Watson, 2019, p. 105). In cases where the information from the abstract was not sufficient, the conclusion section was read as well. Generally, the approach to our screening was inclusive, in the sense of always including studies when in doubt. If there were any discrepancies in the reviewers' assessments, the decision to include or exclude the article(s) was discussed.

Inclusion criteria

To ensure the relevance and quality of the literature selected for our research, we have established specific criteria for the inclusion of articles:

- The article must be peer-reviewed.
- The article must be written in either English or Norwegian.
- The article must address security challenges, risks, or controls within CI/CD environments.

- Preferably, the article should not be older than five years. However, articles older than this may still be considered if they are deemed highly relevant to our research problem and questions.
- Articles may be excluded if they originate from journals or conferences not recognized as level 1 or 2 in the Norwegian register for scientific journals, series, and publishers.

Data extraction, analysis and synthesis

Before we could synthesize the data, we had to complete the data extraction phase. This phase primarily involved coding the material collected during the literature review. It is crucial to determine whether the coding will be inductive or deductive - that is, whether it will be driven by the data itself or by preexisting concepts (Xiao & Watson, 2019). In our case, we chose inductive coding, which relies directly on the data. This approach allowed us to generate insights that are firmly grounded in the reviewed literature.
Furthermore, we relied upon Xiao and Watson's (2019) recommendation that when working as a team, it is beneficial to code a few papers together before dividing the workload. This strategy ensures that all team members have a unified understanding and apply similar standards to the coding process.

After completing the data extraction and coding phase, we proceeded to analyze and synthesize the data to address our supporting research questions, which drives the main research question (Xiao & Watson, 2019). This analysis involved categorizing the data to identify patterns and establish coherent groups, particularly focusing on the focus areas of interest.

In this analytical process, we utilized the methodology outlined by Gioia et al. (2013). This method provided a structured approach to analyzing data, especially useful when dealing with large coding-material. According to Gioia et al. (2013), in the initial categorization phase, we identified numerous categories, 53 in total - which represented different focus areas for assessing the security of the CI/CD pipeline. These were our first-order categories. As we progressed to the second-order theoretical analysis, we examined the similarities and differences among these categories, eventually consolidating them into 10 distinct focus areas. These refined focus areas represented the culmination of our synthesis process, helping us clarify the key aspects of our study.

3.2.3 Grey Literature Review

The concept of Grey Literature (GL) and performing a Grey Literature Review (GLR) encompasses various forms, with definitions provided by different sources. One widely recognized definition, known as the Luxembourg definition, states that grey literature "is produced on all levels of government, academics, business and industry print and electronic format, but which is not controlled by commercial publishers i.e., where publishing is not a primary activity for the producing body" (Schöpfel & Farace, 2009). Another definition, referred to as the Cochrane definition, specifies grey literature as comprising materials that

aren't formally published in conventional sources such as books or journal articles (Lefebvre et al., 2008).

Furthermore, Garousi et al. (2019) classify different types of grey literature based on the model introduced by Adams et al. (2016). This model categorizes grey literature into three tiers, considering two dimensions: expertise and outlet control. "Expertise" refers to the assessability of the creator's credentials and knowledge, ranging from known to unknown. For instance, some grey literature may originate from organizations with clear and authoritative backgrounds, while others may come from less credible sources. Moving to the dimension of "outlet control," which also spans from known to unknown, high outlet control (known) indicates rigorous quality control processes such as peer review and professional editorial oversight before publication (Garousi et al., 2019).

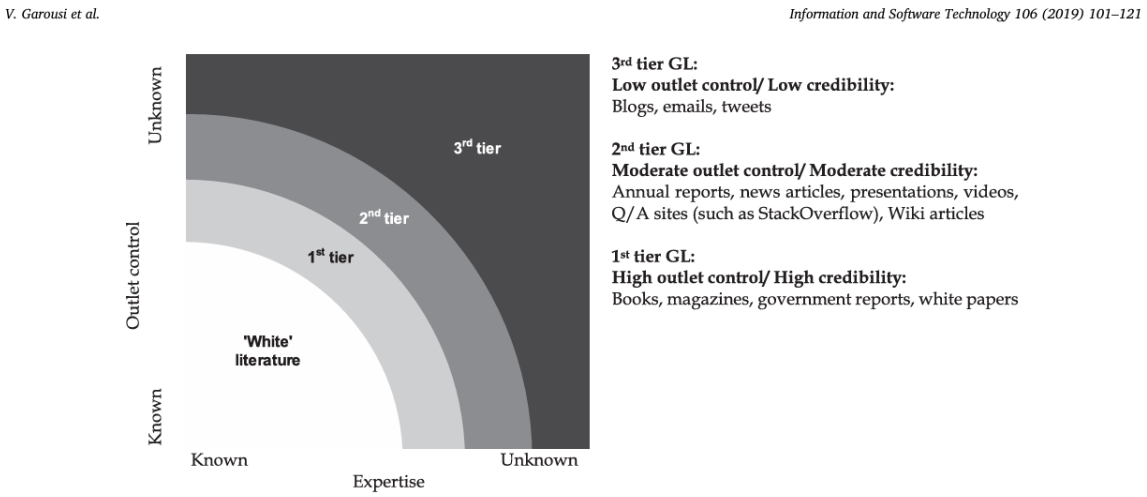

Figure 2: The tiers of grey literature (Garousi et al. (2019))

In contrast, Garousi et al. (2019) emphasize the limited controls over expertise and outlet within grey literature, underscoring the importance of identifying its producers. Moreover, Garousi et al. (2019) reference Giustini (2010), who identified the following producers of grey literature: (1) Government departments and agencies (at municipal, provincial, or national levels), (2) Non-profit economic and trade organizations, (3) Academic and research institutions, (4) Societies and political parties, (5) Libraries, museums, and archives, (6) Businesses and corporations, and (7) Freelance individuals, such as bloggers, consultants, and web 2.0 enthusiasts.

The motivation behind the need for a MLR in our research

Garousi et al. (2019) stress the importance of determining whether to conduct a Systematic Literature Review (SLR), a Grey Literature Review (GLR), or a Multivocal Literature Review (MLR). They provide a checklist to aid in deciding whether to incorporate grey literature into the research, leading to conducting an MLR. This checklist evaluates the complexity of the subject under research, the quantity of evidence available, the level of consensus in the field, and the objectives and practical applications of the research project. Therefore, if one or more "yes" responses are noted, it indicates the inclusion of grey

literature in the research is warranted. Following this, our response along with the rationales for opting to conduct an MLR is provided.

#	Question	Answer
1	Is the subject "complex" and not solvable by considering only the formal literature?	Yes
2	Is there a lack of volume or quality of evidence, or a lack of consensus of outcome measurement in the formal literature?	Yes
3	Is the contextual information important to the subject under study?	Yes
4	Is it the goal to validate or corroborate scientific outcomes with practical experiences?	Yes
5	Is it the goal to challenge assumptions or falsify results from practice using academic research or vice versa?	No
6	Would a synthesis of insights and evidence from the industrial and academic community be useful to one or even both communities?	Yes
7	Is there a large volume of practitioner sources indicating high practitioner interest in a topic?	Yes

Table 4: Checklist to decide whether to include grey literature (Garousi et al., 2019))

3.2.4 Conducting the Grey Literature Review

When planning a Grey Literature Review (GLR), it is important to organize it into distinct phases. For our GLR, we structured it into the following sections:

- Search process: Outlining how and where the information will be gathered.
- When to stop the search: Determining when sufficient data has been collected.
- Source selection: Specifying the criteria for choosing the data sources

Search process

Garousi et al. (2019) pointed out that the search approach for conducting a Grey Literature (GL) search is distinct from that used in a Systematic Literature Review (SLR) within academic databases, where specific search strings are defined. In our search process, we predominantly utilized Google's search engine, creating various search queries and examining the results provided. Additionally, due to our familiarity with different institutional and governmental agencies over time, we had specific sites in mind, such as exploring resources from NIST, among others.

When to stop the search

When conducting a GL-search, determining when to stop isn't always straightforward. Unlike searching in an academic database, where you might receive around 250 articles, a Google search could return anywhere from 100,000 to millions of results. To address this challenge, we turned to Garousi et al. (2019) and their three stopping rules:

1. The first stopping rule is theoretical saturation, where finding more articles doesn't lead to additional insights.

2. The second stopping rule is influenced by the sheer volume of data, such as when a Google search returns 1,000,000 hits.

3. The third stopping rule is reached when you begin encountering varying quality and availability of evidence as you navigate through Google's search results, thus

requiring a degree of trust in the search engine's reliability.

Source selection and quality assessment

Additionally, once potential sources have been obtained, they require further assessment. Therefore, we base our decision-making process on a quality assessment of these sources. This assessment is facilitated by a comprehensive checklist that encompasses various categories and questions to be addressed. It's important to exercise discretion in determining which sources to include and exclude (Garousi et al., 2019). Moreover, our decisions are supported by quantifiable measurements. We compile the responses (0, 0.5 or 1) for each category and question, and then calculate the average by dividing the total score by the number of assessed factors, which in this case is 20 questions (or the numbers of applicable questions).

Criteria	Qs number	Questions
Authority of the producer	1	Is the publishing organization reputable?
	2	Is an individual author associated with a reputable organization?
	3	Has the author published other work in the field?
	4	Does the author have expertise in the area?
Methodology	5	Does the source have a clearly stated aim?
	6	Does the source have a stated methodology?
	7	Is the source supported by authoritative, contemporary references?
	8	Are any limits clearly stated?
	9	Does the work cover specific question?
	10	Does the work refer to a particular population or case
Objectivity	11	Does the work seem to be balanced in presentation?
	12	Is the statement in the sources as objective as possible? Or is the statement a subjective opinion?
	13	Is there vested interest?
	14	Are the conclusion supported by data
Date	15	Does the item have a clearly stated date?
Position w.r.t. related sources	16	Have key related GL or formal sources been linked to/discussed?
Novelty	17	Does it enrich or add something unique to the research
	18	Does it strengthen or refute a current position
Impact	19	Normalize all the following impact metrics into a single aggregated impact metric (when data are available): Number of citations, Number of backlinks, Number of social media shares (the so-called “alt-metrics ”), Number of comments posted for a specific online entries like a blog post or a video, Number of page or paper views

Outlet type	20	1st tier GL (measure = 1): High outlet control/ High credibility: Books, magazines, theses, government reports, white papers 2nd tier GL (measure = 0.5): Moderate outlet control/ Moderate credibility: Annual reports, news articles, presentations, videos, Q/A sites (such as StackOverflow), Wiki articles 3rd tier GL (measure = 0): Low outlet control/ Low credibility: Blogs, emails, tweets

Table 5: Checklist for quality assessment of grey literature (Garousi et al., 2019))

3.2.5 MLR – CI/CD Security Focus Areas

Systematic Literature Review

For our Systematic Literature Review (SLR), we searched for relevant academic literature using electronic databases such as Web of Science, IEEE Xplore, and Scopus. Additionally, we conducted both backward and forward searches based on the literature selected during the screening process. This comprehensive approach ensures that we capture a wide range of articles, thereby enhancing the depth and breadth of our review.

After formulating the research question and conducting some preliminary searches, we compiled a list of relevant keywords. These keywords were then utilized to construct search queries aimed at identifying pertinent literature.

Keyword	**Synonyms**
CI/CD	Continuous integration Continuous delivery Continuous deployment Continuous practices CI/CD pipeline CI pipeline CD pipeline DevOps pipeline DevSecOps pipeline SecDevOps pipeline
Security	Cybersecurity Cyber security
Challenge	Challenges Barrier(s) Problem(s) Vulnerability Vulnerabilities Defect(s) Risk(s) Flaw(s) Threat(s) Attack(s) Breach(es)

Table 6: Keywords and synonyms for focus area SLR search queries

CI/CD focus areas search string

Furthermore, to carry out the search, we needed to define a search query to use against the academic literature databases. Our search incorporated the keywords "CI/CD," "Security," and "Challenges," along with synonyms for each keyword.

Query	WoS	Scopus	IEEE Xplore
("CI/CD" OR "Continuous integration" OR "Continuous delivery" OR "Continuous deployment" OR "Continuous practices" OR "CI/CD pipeline*" OR "CI pipeline*" OR "CD pipeline*" OR "DevOps pipeline*" OR "DevSecOps pipeline*" OR "SecDevOps pipeline*") AND (Security OR Cybersecurity OR "Cyber security") AND (Challenge OR Challenges OR Barrier OR Barriers OR Problem OR Problems OR Vulnerability OR Vulnerabilities OR Defect OR Defects OR Risk OR Risks OR Flaw OR Flaws OR Threat OR Threats OR Attack OR Attacks OR Breach OR Breaches)	123	239	266

Table 7: Focus area SLR search results

Screening

After completing the search and extracting all relevant articles, we screened these articles to determine whether to include them for data extraction and further analysis. Moreover, this screening process was conducted using the three-stage procedure proposed by Xiao & Watson (2019), where we first reviewed the titles and abstracts, followed by a full-text review.

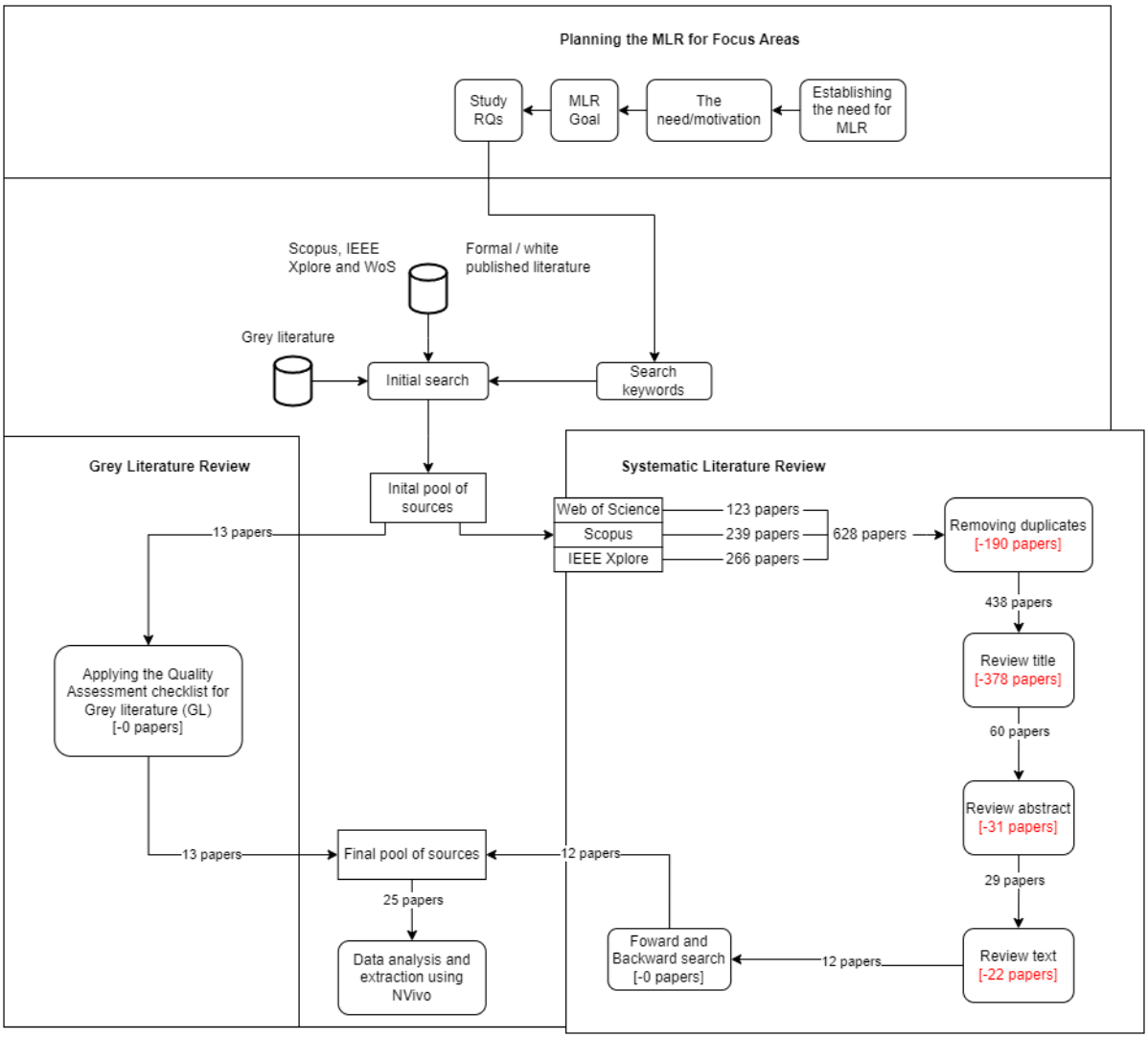

Figure 3: Focus area MLR screening

Systematic literature review - Articles

Following the completion of the systematic literature review, a total of 12 articles were obtained. These articles were subsequently utilized during the data extraction and analysis phase in NVivo, demonstrating its contribution.

Reference	Title
Pecka et al. (2022)	Privilege Escalation Attack Scenarios on the DevOps Pipeline Within a Kubernetes Environment
Koishybayev et al. (2022)	Characterizing the Security of Github CI Workflows
Pan et al. (2024)	Ambush From All Sides: Understanding Security Threats in Open-Source Software CI/CD Pipelines
Moriconi et al. (2023)	Reflections on Trusting Docker: Invisible Malware in Continuous Integration Systems
Bajpai & Kannavara (2023)	Misplaced Trust: The Security Flaw in Modern Code Signing Process
Ahmadvand et al. (2018)	Integrity Protection Against Insiders in Microservice-Based Infrastructures: From Threats to a Security Framework

Rafi et al. (2020)	Prioritization Based Taxonomy of DevOps Security Challenges Using PROMETHEE
Akbar et al. (2022)	Toward successful DevSecOps in software development organizations: A decision-making framework
Rajapakse et al. (2022)	Challenges and solutions when adopting DevSecOps: A systematic review
Zhou et al. (2023)	Revisit security in the era of DevOps: An evidence-based inquiry into DevSecOps industry
Shahin et al. (2017)	Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices
Gu et al. (2023)	Continuous Intrusion: Characterizing the Security of Continuous Integration Services

Table 8: Included articles - Focus area SLR

Grey literature Review – Focus Areas

When conducting the search within Grey Literature (GL), we primarily used Google to perform searches with specific keywords, such as "CI/CD Security" and "CI/CD Security Challenges," which were identified from a table of keywords and synonyms used in the SLR. Following the approach recommended by Garousi et al. (2019), we primarily relied on the Google search engine to explore the results. Additionally, we reviewed sources from various governmental institutions like NIST, CISA, and NSA, leveraging our prior experience with these agencies. This search process resulted in the identification of 13 articles that supplemented the literature from the Systematic Literature Review (SLR). We concluded the search at this point due to reaching theoretical saturation and the substantial volume of data retrieved from the Google search.

Grey literature review - Articles

ID	Reference	Title
GL1	The Hacker News (2023)	CI/CD Risks: protecting Your Software Development Pipelines.
GL2	Morgnestern (2023)	CI/CD security – 5 best practices.
GL3	Codefresh (n.d.)	CI/CD Security: 7 Risks and What you Can Do About Them
GL4	Dancuk (2021)	CI/CD Security – How to Secure your CI/CD Pipeline
GL5	Sysdig (n.d.)	CI/CD Security: Securing Your CI/CD Pipeline
GL6	Maayan (n.d.)	DevOps Security Challenges and How to Overcome Them
GL7	Brukman (2023)	DevOps threat matrix
GL8	Kulanov & Stepanov (2023)	Elevating CI/CD Security With Supply Chains
GL9	Chau et al. (2023)	Getting started with CI/CD pipeline security
GL10	Palo Alto Networks (n.d.)	What Is the CI/CD pipeline
GL11	Krivelevich & Gil (2022)	OWASP Top 10 CI/CD Security Risks
GL12	NSA & CISA (2023)	Defending Continuous Integration/Continious Delivery (CI/CD) Environments

Table 9: Included articles - Focus area GLR

Source selection and quality assessment

In selecting sources, we focus on the grey literature screening framework as outlined by Garousi et al. (2019). We screened 12 grey literature papers, each receiving an average normalized score of 0.79 based on their alignment with the established screening criteria. For a detailed analysis of the screening assessment, see Appendix D. Consequently, all these papers were included in the final source pool.

3.2.6 MLR – CI/CD Security Practices

Systematic Literature Review

For our systematic literature review (SLR) on CI/CD security practices, we utilized the same electronic databases as in our previous SLR, specifically Web of Science, IEEE Xplore, and Scopus. Our goal was to identify academic literature that defines security practices within a CI/CD context. Additionally, after the initial screening process, we conducted both backward and forward citation tracking.

However, before initiating the search, we compiled a list of relevant keywords and their synonyms. These were used to construct a comprehensive search string for the literature review.

Keyword	Synonyms
CI/CD	Continuous integration Continuous delivery Continuous deployment Continuous practices DevOps DevSecOps CI/CD pipelines CI pipeline CD pipeline DevOps pipeline DevSecOps pipeline SecDevOps pipeline Delivery pipeline Development pipeline
Security	Cybersecurity
Security practices	Best practice Best practices

Table 10: Keywords and synonyms for security practices SLR search queries

CI/CD Security practices search string

To conduct the search, we needed to develop a search string to effectively retrieve academic literature on CI/CD security practices. This search string included the keywords "CI/CD," "Security," and "Security Practices."

Query	WoS	Scopus	IEEE Xplore
("CI/CD" OR CICD OR "Continuous integration" OR "Continuous delivery" OR "Continuous deployment" OR "Continuous practices" OR DevOps OR DevSecOps OR "CI/CD pipeline*" OR "CI pipeline*" OR "CD pipeline*" OR "DevOps pipeline*" OR "DevSecOps pipeline*" OR "SecDevOps pipeline*" OR "Delivery pipeline" or "Development pipeline") AND (Security OR Cybersecurity) AND ("Security practice*" OR "Best practice" OR "Best practices")	106	240	151

Table 11: Security practices SLR search results

Screening

After completing our search and extracting all relevant articles on the topic of CI/CD security practices, we applied the Xiao & Watson (2019) three-stage procedure to further screen these articles. This involved reviewing and assessing the titles, abstracts, and full texts of the articles for data extraction.

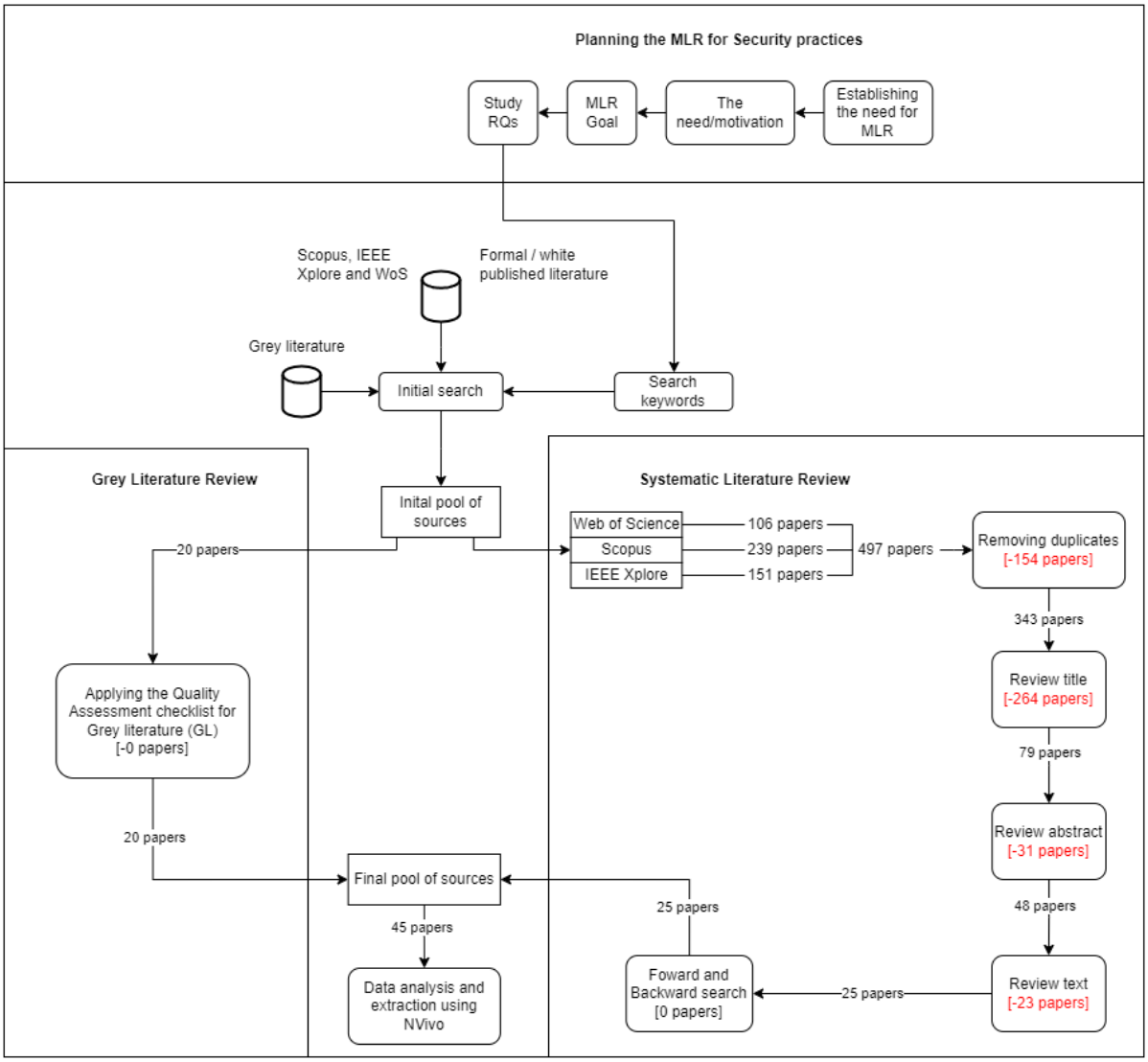

Figure 4: Security practices MLR screening

Systematic Literature Review – Articles

After conducting a systematic literature review on CI/CD security practices, we identified 25 relevant papers, which were included in our final pool of sources. Subsequently, during the data extraction and analysis phase, three articles were excluded from consideration due to minimal usage in our research.

Reference	Title
Vasile et al. (2019)	Applying Security Concepts to Continious Integrations for the Purpose of Testing Embedded Systems
Rajapakse et al. (2022)	Challenges and solutions when adopting DevSecOps: A Systematic review
Nalini et al. (2023)	CI/CD Pipeline with Vulnerability Mitigation
Rangnau et al. (2020)	Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines
Hastings & Walcott (2022)	Continuous Verification of Open Source Components in a World of Weak Links
Larios-Vargas et al. (2022)	DASP: A Framework for Driving the Adoption of Software Security Practices.
Patra et al. (2022)	Docker Security: Threat Model and Best Practices to Secure a Docker Container
Angermeir et al. (2021)	Enterprise-Driven Open Source Software: A Case Study on Security Automation
Ahmadvand et al. (2018)	Integrity Protection Against Insiders in Microservice-Based Infrastructures: From Threats to a Security Framework
Neharika & Lennon (2023)	Investigations into Secure IaC Practices
Bajpai & Kannavara (2023)	Misplaced Trust: The Security Flaw In Modern Code Signing Process.
Zeini et al. (2023)	Preliminary Investigation into a Security Approach for Infrastructure as Code
Vakhula et al. (2023)	Research on Security Challenges in Cloud Environments and Solutions based on the “Security-as-Code” Approach
Bajpai & Lewis (2022)	Secure Development Workflows in CI/CD Pipelines
Rahman et al. (2021a)	Shhh!: 12 Practices for Secret Management in Infrastructure as Code
Martínez & Durán (2021)	Software Supply Chain Attacks, a Threat to Global Cybersecurity: SolarWinds Case Study
Shevchuk et al. (2023)	Software for Improve the Security of Kubernetes-based CI/CD Pipeline
Leppänen et al. (2022)	Trends for the DevOps Security. A Systematic Literature Review
Tak et al. (2017)	Understanding Security Implications of Using Containers in the Cloud
Martin (2020)	Visibility & Control: Addressing Supply Chain Challenges to Trustworthy Software-Enabled Things.
Kumar & Goyal (2021)	When Security Meets Velocity: Modelling Continuous Security for Cloud Applications using DevSecOps
Shamim et al. (2020)	XI Commandments of Kubernetes Security: A Systematization of Knowledge Related to Kubernetes Security practices.

Table 12: Included articles - Security practices SLR

Grey literature Review – CI/CD Security Practices

In the exploration of Grey Literature (GL) for this Multivocal Literature Review (MLR), our search strategy involved using Google to locate relevant articles by applying specific keywords identified from a predefined list that included terms like "CI/CD Security" and "CI/CD Security practices." This method aligns with the practices suggested by Garousi et al. (2019) for effectively utilizing search engines. We also examined documents from authoritative bodies such as NIST, CISA, and NSA, capitalizing on our previous

engagements with these entities. Our search efforts yielded 20 articles that enriched the findings from the Systematic Literature Review (SLR). The search was concluded once theoretical saturation was achieved and a significant amount of data had been collected through Google.

Grey literature review – Articles

ID	Reference	Title
GL1	Boote et al. (2023)	BSIMM14 Report 2023
GL2	CIS (2022)	CIS Software Supply Chain Security Guide
GL3	CNCF (2021)	Software Supply Chain Best Practices
GL4	NSA & CISA (2023)	Defending Continuous Integration/Continuous Delivery (CI/CD) Environments
GL5	NSA et al. (2023)	Securing the Software Supply Chain: Recommended Practices for Managing Open-Source Software and Software Bill of Materials
GL6	Microsoft (n.d.)	What are the Microsoft SDL practices?
GL7	Souppaya et al. (2017)	NIST SP 800-190: Application Container Security Guide
GL8	Chandramouli et al. (2024)	NIST SP 800-204D: Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines
GL9	Souppaya et al. (2022)	NIST SP 800-218: Secure Software Development Framework (SSDF)
GL10	Scovetta (2020)	Threats, Risks, and Mitigations in the Open Source Ecosystem
GL11	OWASP (n.d.)	CI/CD Security Cheat Sheet
GL12	Krivelevich & Gil (2022)	OWASP Top 10 CI/CD Security Risk
GL13	Yazdani & Thakur (n.d.)	OWASP DevSecOps Guideline
GL14	Pagel & Prasad (n.d.)	OWASP DSOMM
GL15	Deleersnyder & Win (n.d.)	OWASP SAMM Version 2
GL16	Springett (2020)	OWASP SCVS Version 1.0
GL17	Diglio & Wang (2023)	Secure Supply Chain Consumption Framework (S2C2F)
GL18	Ng et al. (2022)	Securing the pipeline and CI/CD workflow
GL19	Supply-chain Levels for Software Artifacts (n.d.)	Get started – Choosing your SLSA level.
GL20	Moghnie et al. (2020)	The Six Pillars of DevSecOps: Automation

Table 13: Included articles - Security practices GLR

Source selection and quality assessment

Based on the grey literature screening framework outlined by Garousi et al. (2019), we reviewed 20 grey literature documents. Each document received an average normalized score of 0.92, based on its alignment with the established screening assessment criteria. For a more detailed overview, see Appendix E. As a result, all articles were included in the source pool.

3.2.7 MLR – Existing CI/CD Security Maturity Models

Systematic Literature Review

For our systematic literature review (SLR) on existing Maturity Models (MM), we used the same electronic databases as in our previous SLR, namely Web of Science, IEEE Xplore, and Scopus. Our objective was to deepen our understanding of the published literature and to

determine the presence of any security-oriented maturity models. We also aimed to explore how these models could be integrated with our research problem.

Before beginning the search, we developed a list of relevant keywords and their synonyms, which we used to formulate a comprehensive search string for the literature review. Additionally, after the initial screening process, we employed both backward and forward citation tracking to further enrich our search.

Keyword	Synonyms
CI/CD	CICD Continuous integration Continuous delivery Continuous deployment Continuous practices DevOps DevSecOps SecDevOps
Security	Cybersecurity Cyber security IT security
Maturity model	Capability model Framework

Table 14: Keywords and synonyms for existing maturity models SLR search queries

Existing CI/CD Security Maturity Model – Search String

To conduct the search, we needed to develop a search string to effectively retrieve academic literature on existing CI/CD maturity models. This search string included the keywords "CI/CD," "Security," and "Maturity model."

Query	WoS	Scopus	IEEE Xplore
("CI/CD" OR cicd OR "Continuous integration" OR "Continuous delivery" OR "Continuous deployment" OR "Continuous practices" OR devops OR devsecops OR secdevops) AND "*Security*" AND ("Maturity model" OR "Capability model" OR framework)	73	162	128

Table 15: Existing maturity models SLR search results

Screening

After completing our search and extracting all relevant articles on the topic of existing CI/CD maturity models, we applied the Xiao & Watson (2019) three-stage procedure to further screen these articles. This involved reviewing and assessing the titles, abstracts, and full texts of the articles for data extraction.

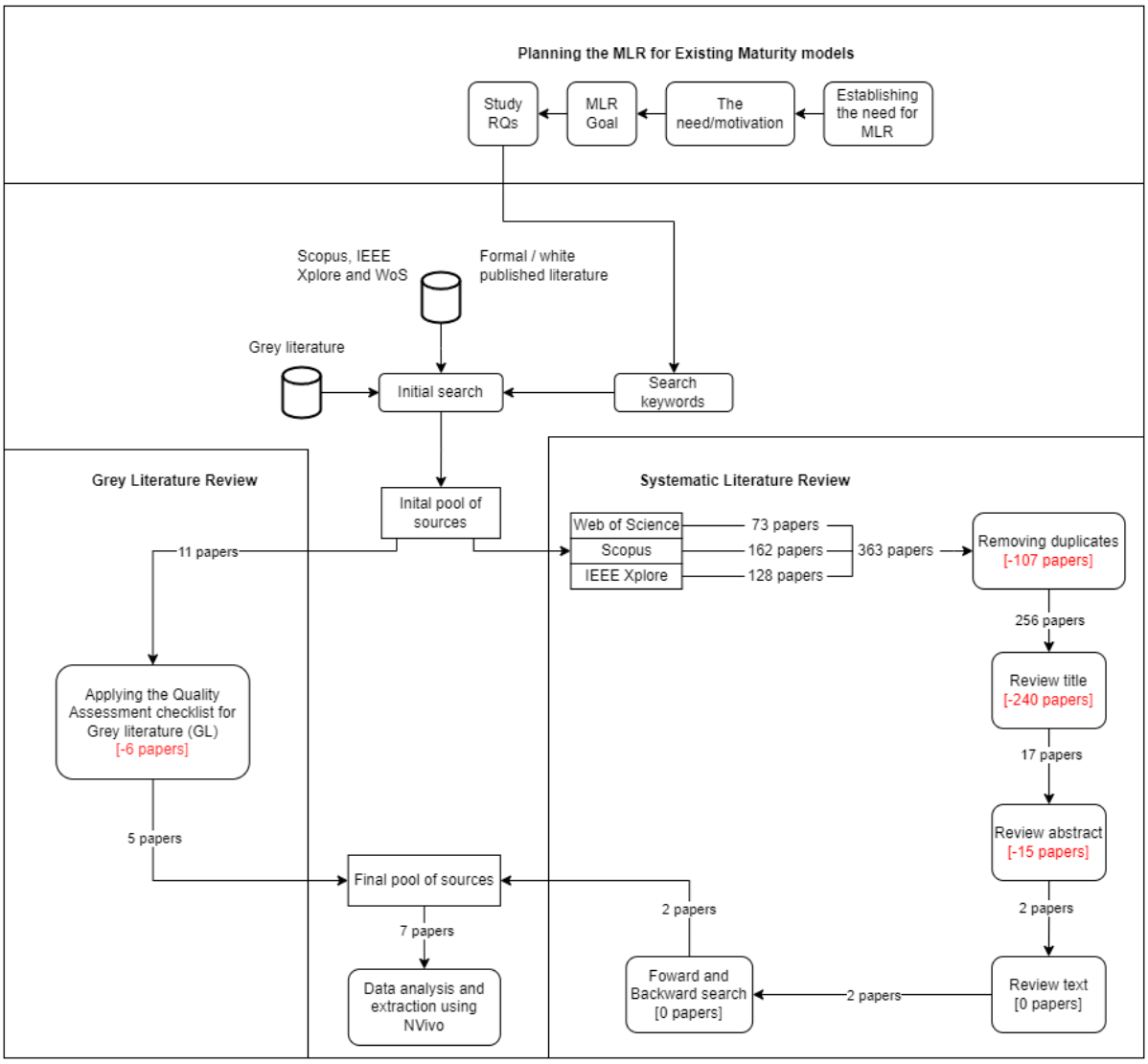

Figure 5: Existing maturity models MLR screening

Systematic Literature Review – Articles

Authors	Title
Kumar et al. (2023)	Prioritization of DevOps Maturity models using Fuzzy TOPSIS
Brasoveanu et al. (2022)	Security Maturity Self-Assessment Framework for Software Development Lifecycle

Table 16: Included articles - Existing maturity models SLR

Grey Literature Review – CI/CD Existing Security Maturity Models

For the Multivocal Literature Review (MLR) on Grey Literature (GL), we developed a search strategy that utilized Google to identify pertinent articles. We applied specific keywords from a predefined list, which included terms such as "CI/CD Security" and "Maturity Models," following the guidelines recommended by Garousi et al. (2019) for efficient search engine use. Additionally, we reviewed documents from established organizations like NIST, CISA, and NSA, building on our previous interactions with these groups. Our search efforts resulted in the discovery of 11 articles that augmented the insights gathered from the Systematic

Literature Review (SLR). We concluded our search upon reaching theoretical saturation and collecting a substantial amount of data via Google.

Grey literature review – Articles

ID	Reference	Title
GL1	Vlietland (2019)	Continuous Delivery 3.0 Maturity Model (CD3M)
GL2	Rehn et al. (2013)	The Continuous Delivery Maturity Model
GL3	Hornbeek & Jones (n.d.)	A Roadmap to Continuous Delivery Pipeline Maturity
GL4	Pagel & Prasad (n.d.)	The OWASP DevSecOps Maturity Model (DSOMM)
GL5	Veritis (n.d.)	DevOps Maturity Model

Table 17: Included articles - Existing maturity models GLR

Source selection and quality assessment

It's crucial to note that despite some articles achieving a score above our predefined threshold, they were still excluded due to their lack of relevance. This decision was reflected in our assessment scores, where articles that did not provide unique insights were assigned a zero in questions 17 and 18 of our screening assessment. See Appendix F for a detailed overview of the screening. Ultimately, five articles were selected from the Grey Literature Review (GLR).

3.3 Environment

In this subsection, the environment is described by presenting the data collection of the project. The informants constitute the environment which has contributed to the design and evaluation of the artifact. The recruitment of informants aimed at reflecting the breadth in roles and organizations of the environment which the artifact will be applied in.

Eleven informants (Table 18) from eight different organizations (Table 20) were interviewed during the design iterations. Additionally, two more informants were contributing to the project. Informant 12 provided written feedback via email in iteration 3, and informant 13 participated in the case study evaluation. The recruitment of informants was a combination of using our professional networks, searching and contacting people on LinkedIn, and recommendations from our supervisors. Several of the informants were selected because of their engagement and contributions to the software development and/or information security communities. As an illustration, at least 6 of the informants have been speakers at conferences, contributed to white papers or other forms of informative artifacts, or established/organized meeting places and interest groups. The majority of the informants worked in Norway, but some worked in Finland and USA.

Pseudonym	Role	Involvement
Interviewee 1	Cloud Solution Architect	Iteration 1
Interviewee 2	Cloud Native Engineer	Iteration 1
Interviewee 3	Cloud Native Architect	Pilot interview Iteration 1
Interviewee 4	Security Architect	Iteration 1
Interviewee 5	Cloud Advisor	Iteration 1
Interviewee 6	Chief Technology Officer	Iteration 2

Interviewee 7	Software Engineer and Business Developer within secure development	Iteration 2
Interviewee 8	Platform Engineer	Iteration 2 Iteration 3 (emails) Case study evaluation
Interviewee 9	Lead Developer	Iteration 2
Interviewee 10	Lead IT Architect	Iteration 2
Interviewee 11	Manager and subject matter expert in secure development	Iteration 3
Informant 12	Security Consultant	Pilot interview Iteration 3 (emails)
Informant 13	Platform Security Engineer	Case study evaluation

Table 18: Overview of informants

Ten semi-structured interviews were conducted in the iterative design process of the project, with one of them being a group interview with two informants from the same organization. The interviews lasted between 45 to 90 minutes, depending on the amount of feedback from the informants. The number of practices in the model was always between 170 and 216, so we did not have the time to go through each practice in most of the interviews. Ahead of the interview, the informants received the draft of the model and were asked to select at least three focus areas to review before the interview. See Appendix G for an overview of which areas the informants reviewed.

When	**Focus of the phase**	**Interviews**
Iteration 1	Focus areas and sub-areas	5
Iteration 2	Practices' validity, relevance, and maturity level	4 (1 group interview)
Iteration 3	Control questions and self-assessment	1 (plus 4 emails)
Case study evaluation	Evaluating the relevance and utility of the MM	1

Table 19: Overview of phases of the project and number of interviews per phase

Organization	**Sector / Type**	**Country**
Organization 1	Technology	Multinational (Norwegian office)
Organization 2	Consultancy	Multinational (Norwegian office)
Organization 3	Technology	USA
Organization 4	Consultancy	Multinational (Norwegian office)
Organization 5	Consultancy	Finland
Organization 6	Finance	Norway
Organization 7	Telecom	Norway
Organization 8	Consultancy	Norway

Table 20: Overview of organizations

At the end of the project, a case study evaluation was conducted. The naturalistic evaluation (Venable et al., 2016) session was conducted with two participants from one of the organizations that were involved in the iterative design process. There were several reasons why this organization was selected for the evaluation. This organization, and the

representants participating, was willing and available for doing the case study. The organization was considered as a relevant case since it is a large organization with many software development teams. Even though the organization's primary product/service is not software, all of their internal and external services are dependent on software. Additionally, security is critical for this organization since the organization is an Operator of Essential Services (OES) (NIS1 Directive, 2016).

3.4 Build and evaluate

Based on Hevner et al.'s (2004) seven guidelines for design science research, Becker et al. (2009) have established eight requirements for development of maturity models. The first requirement is conducting a comparison of the proposed model with existing maturity models. This will assure that any potential model fully or partially addressing the problem to be solved is identified. When comparing with existing models, there may be some aspects or elements of the models that can be implemented into or used as inspiration for the proposed model. In some cases, it may be sufficient to just modify an existing model to make it relevant to the identified problem to be solved. In our project, none of the identified existing models covered CI/CD security specifically enough. Nor would it be expedient to expand or modify any of the existing models. However, some of the content of the models was used as inspiration when we defined focus areas and practices for our maturity model.

The second requirement is developing the maturity model iteratively. The timeframe of the master thesis project was limited to approximately five months, but we managed to arrange the work into iterations. Due to the fact that the time was limited, the iterations were time framed to fit interviewing up to five experts and make adjustments to the artifact based on the feedback. Thus, the iterations were not stopped by a defined "trigger", but rather planned to ensure progression. A total of 4 iterations were conducted in our design of the CI/CD security maturity model, where the last iteration consisted of a naturalistic evaluation of the artifact.

Evaluation is the third requirement. Additionally, the evaluation must be done iteratively as well. In our project, the interviews in each iteration functioned as evaluation, since the amount of feedback concerning adjustments of the model or aspects such as the quality, usefulness, and effectiveness of the model would indicate the perceived relevance and usefulness of the artifact. Furthermore, a case study evaluation was performed at the end of the project. Evaluation goals were defined to have predetermined goals to compare the model against (table 21). All of the evaluation goals reflect the goals we worked towards during the design process, but formally these goals were only used for the summative evaluation. The evaluation goals are inspired by the evaluation goals of the Extended Zero Trust Maturity Model presented by Tokerud et al. (2023). The aim when defining the goals was to end up with a set of goals that could measure the fulfillment of the requirements and constraints of the problem to solve. Fulfillment of such goals would indicate a complete and effective design artifact (Hevner et al., 2004).

Evaluation goal	Criteria for fulfillment
The model can be used to self-assess the current CI/CD security maturity of organizations/teams	The evaluands can independently use the self-assessment, without relying heavily on assistance from the evaluators The results from the self-assessment reflects the current maturity well
The model is applicable for organizations with CI/CD pipelines	The evaluands consider the artifact to be relevant. The evaluands perceive the components of the artifact to fit the context of its organization
The model can be used to improve organizations'/teams' security capabilities within CI/CD	The evaluands get an understanding of which capabilities they are lacking to achieve a higher maturity. Improvement initiatives grounded on the use of the artifact result in a better CI/CD security
The security practices in the model are placed in appropriate levels, relevant, and comprehensible	The evaluands consider the security practices to be relevant The evaluands understand what the practices imply The evaluands consider the progression of practices throughout the levels to be reasonable
The informants which get to see the model show an enthusiasm for the model and want to adopt it into their organization	←
The informants suggest few or no modifications to the latest draft	←

Table 21: Evaluation goals for the maturity model

Developing the maturity model employing a multi-methodological approach is the fourth requirement. Our project did also comply with this requirement, as we utilized literature reviews, interviews, and a case study throughout the project.

The fifth and sixth requirements are related to the second guideline (problem relevance) by Hevner et al. (2004). The problem must be defined (requirement 6), and the relevance of the problem solution (maturity model) must be identified and demonstrated (requirement 5). Before the literature reviews and the design iterations were conducted, we defined the problem as a twofold problem. The number of cyber security incidents where CI/CD is exploited has increased, and there is a lack of artifacts that enable easy and effective assessment and enhancement of the security posture within CI/CD of an organization or a software development team. The relevance of the problem and solution was identified both through the increasing exploitation of CI/CD infrastructure and the informants' confirmation of the relevancy of the problem and our maturity model when they were interviewed.

The presentation of the maturity model has to be targeted towards the needs of its users and the conditions of its application, according to the seventh requirement. The Excel file consisting of the self-assessment questionnaire and a maturity dashboard is presented

accordingly to the users' needs, utilizing the terminology used by the practitioners and guiding the users on how to use the spreadsheet.

Scientific documentation is the last requirement. This master thesis constitutes the scientific documentation of the design process, involved parties, applied methods, and results of the project.

Maturity models have been subject for critique when it comes to the documentation of the design process. The critique can be divided into several aspects, such as a lack of quality documentation of the design process (or any documentation at all) and designing a MM as the researchers chose – without following a verifiable approach (Adekunle et al., 2022; Becker et al., 2009). Thus, we synthesized a maturity model design process consisting of 4 common phases identified in the past literature describing maturity model design methodologies (A comparison of these can be found in Appendix C). The process starts with a preparative phase, followed by a design phase which is conducted before the evaluation phase. Ultimately, the process ends with the deployment and reporting phase.

Common phases	Activities/principles
Preparative phase	• Understanding and scoping the problem domain • Identifying the need for the proposed artifact • Compare existing maturity models within the domain • Determine a design strategy • Identify stakeholders that can assist in the development
Design phase	• Iterative design process • Identify relevant components for the maturity model (multi-methodological approach) • Formulate control questions • Design a self-assessment questionnaire
Evaluation phase	• Formative evaluation through expert interviews • Summative evaluation through a naturalistic case study
Deployment and reporting phase	• Communicate the design results to practitioners and the scientific community

Table 22: Overview of what each phase involved in our project

Preparative phase

In our project, the preparative phase involved understanding and scoping the problem domain (CI/CD security). These activities were mentioned in all of the methodologies we reviewed (Becker et al., 2009; de Bruin et al., 2005; Lahrmann et al., 2011; van Steenbergen et al., 2010).

Through literature reviews, pilot interviews, and comparing existing maturity models within security in DevOps/DevSecOps, we identified the need for an artifact that can facilitate maturation within CI/CD security. Identifying and comparing existing maturity models within the same or similar domains is considered as important to be able to determine the strategy of the design (Becker et al., 2009; van Steenbergen et al., 2010). Based on the maturity models we identified, we decided to follow a design strategy of designing a completely new model, which is one of the four basic design strategies described by Becker

et al. (2009). The identified models were not considered to be relevant to be enhanced, or combined into a new model. However, some of the identified models were used in the design process to identify security practices for some of the focus areas of our completely new model.

The preparative phase was also used to identify potential participants for the expert interviews in the design phase. This aligns with one of the major decisions de Bruin et al. (2005) describe for the scoping of the model, which is to identify stakeholders that can assist in the development of the maturity model.

Design phase

The design phase of our project followed an iterative approach. As previously described, this is one of the requirements for the development of maturity models formulated by Becker et al. (2009). The iterations had sub-steps for selecting the focus of the iteration, selecting the approach, designing the section in scope of the iteration, and testing the results. These are the recommended sub-steps of the design iterations, according to Becker et al. (2009).

Focus areas, sub-areas, and capabilities/practices for the maturity model were identified through reviewing the existing knowledge base and qualitatively collecting data through interviews with experts. This adheres to the existing knowledge within maturity model design, which recommends combining methods such as literature reviews and exploratory methods (e.g. interviews) to identify the relevant components for the model (Becker et al., 2009; de Bruin et al., 2005; van Steenbergen et al., 2010). In addition to having focus areas (dimensions), we divided each focus area into sub-areas (sub-dimensions) which de Bruin et al. (2005) recommend for complex domains.

For the measurement of maturity, control questions based on the focus areas and the practices were formulated. A self-assessment questionnaire was filled with these questions, for a convenient solution for measuring maturity. Using control questions and questionnaires for maturity assessments is recommended by both de Bruin et al. (2005) and van Steenbergen et al. (2010).

Evaluation phase

For each iteration in the project, the expert interviews were used to evaluate the drafts of the maturity model and receive feedback on potential improvements. As part of the last iteration, a case study evaluation of the maturity model was conducted to check the model's conformity to the evaluation goals set to it. Thus, the expert interviews were used as formative evaluation of the artifact, while the case study evaluation was used as a summative evaluation of the artifact (Venable et al., 2016).

The evaluation was conducted since it is a crucial step of design science research projects, by critically examining the model's utility, efficacy, reliability, quality, generalizability, and validity, which directly influences the acceptance of the artifact (Hevner et al., 2004; Lahrmann et al., 2011). In the context of designing maturity models, it is important that the

users of the model can be confident that the introduced capabilities really will result in improvement (Helgesson et al., 2012). Furthermore, they should be confident that there are not other capabilities that would result in significantly more value. From this reason, Helgesson et al. (2012) emphasize the importance of being able to show that the MM guides the user to the right improvements. They admit that this requires much time and effort, since it involves empirically investigating a large enough set of improvement initiatives. This is the reason why the evaluation of the designed maturity model from our project did not evaluate the efficacy of improving capabilities.

Deployment and reporting phase

Several of the activities that can be considered as a part of this phase, such as maintenance and regular evaluation, was out of scope for this project. The project ended with deployment of the maturity model and the submission of this master's thesis, eventually resulting in the thesis being published.

The communication of the design results to practitioners and the scientific community (Becker et al., 2009; van Steenbergen et al., 2010) is done with this thesis. If we had more time, the practitioners could have got their own document with more targeted information by excluding the academic details which are required in a master's thesis. However, the Excel-file for the maturity model includes information for guiding the practitioners on the usage of the model.

3.4.1 Data analysis

For analysing the data collected throughout the design iterations, the interviews were recorded (with the informants' consent), transcribed, and subsequently coded using NVivo to organize and centralize the responses to the questions. All feedback that could be used to modify and potentially improve the model was coded to different codes, depending on what aspect of the model the feedback was directed towards. In addition to assembling the feedback in NVivo, we populated an Excel-sheet with the same structure of focus areas and sub-areas of the draft, with columns for each informant's feedback. This gave a good oversight and a good foundation for comparing the opinions of the informants.

The decision to implement suggested improvements was influenced by several factors. First, the consensus of the feedback was considered. For example, if one respondent's comment was contradicted by five others, a compromise was made between these differing views. However, in cases where feedback came from a single interviewee, it was included if relevant to our scope. Second, the applicability of the feedback was assessed based on whether it fell within the project's defined scope. Third, practical constraints such as time and resources were key considerations. For instance, while automating the entire self-assessment process was an appealing suggestion, the constraints of our master thesis project timeline made it unfeasible to implement.

4 The CICDSecMM (CI/CD Security Maturity model)

This section will describe the development process of the CICDSecMM, presenting the iterations that have shaped its current form. The maturity model's initial iteration was developed by reviewing the existing knowledge base and creating a preliminary draft. Following this, three further iterations were conducted. These involved interviewing industry experts and incorporating their insights to refine and enhance the model, ensuring it is highly relevant and applicable to the industry. One last iteration was conducted to perform a case study evaluation of the artifact. At the end of the section, we present the final version of the CI/CD Security Maturity Model.

4.1 Reviewing the knowledge base and creating the foundation of the artifact

The first phase of the project was mainly focused around gaining awareness and knowledge within the problem area of CI/CD security. During the five months before the project officially started, a systematic literature review within CI/CD security was conducted alongside 3 pilot interviews with experienced practitioners within cloud, systems development, and security. From the literature review and the pilot interviews, we found that this area does not have any existing (fully covering) maturity models and that the practice in the field in many cases could benefit from enhancing the maturity.

When the project officially started, two multivocal literature reviews (MLR) were conducted to identify focus areas and practices/capabilities. The findings from the MLRs were used to populate an initial draft of the maturity model, as a matrix/grid consisting of focus areas, sub-areas, and practices placed into maturity levels. The practices were placed on a scale of three maturity levels with ascending maturity. Additionally, two higher levels, 4 and 5, were added but left empty. Level 4 and 5 were mainly added to see what reactions we would get in the interviews, and whether or not the experts would expand the scale or deem the three-level scale sufficient. When the initial draft was put together, it was sent to the interviewees of the first round of interviews in advance of the meeting. This initial draft consisted of 10 focus areas, 39 sub-areas, and 200 practices.

4.2 Iterative development of the maturity model

The next phase consisted of iterative development of the maturity model, with slightly different focus for each iteration. Throughout the iterations, the informants provided feedback and suggested adding, removing, and modifying the focus areas, sub-areas, and practices in the draft reviewed in the respective iteration.

4.2.1 Iteration 1 – Validating the focus areas and sub-areas

The first iteration focused on the focus areas and the sub-areas, even though the practices and maturity levels also were discussed in the interviews. A total of 5 individuals were interviewed in this iteration. Their feedback was used to modify the model to make it more valid within the scope of CI/CD security. The empty maturity levels 4 and 5 were discussed in the interviews, and the interviewees pointed out some practices which could have been

placed at a higher level. Nevertheless, the common perception among the experts was that most of the practices fit well within the scale of three levels.

Interviewee 1 asked if threat modeling was included in the model, and when we said that it was not implemented, he responded that "*it is a critical thing to include*". Thus, we included threat modeling in the model, but since we were not sure where it would fit, we made a row for what we called "orphan areas". This row was made for aspects we wanted some input on where to place in the model.

Both interviewee 3 and interviewee 5 said that key vaults should be used for storing any kind of secrets, not only keys. This feedback led to our reflection on the usage of terms within the focus area for secrets. As a result, the sub-area "key management" was renamed "secret management", and the practice mentioning key vaults was aimed at secrets instead of keys. Within the focus area for secrets, interviewee 1 suggested making a separate sub-area for certificates because *"in our assessments, we distinguish between secrets and certificates"*. We decided to add certificates as a new sub-area for secrets, but we did not have any practices to fill it with. Thus, we left it empty and waited for the next round of interviews to get ideas and feedback for this sub-area. Initially, we also had a sub-area for encryption of secrets, but interviewee 5 meant that *"this is a little bit more about how the data is at rest, not when the data is being used by the CI/CD pipeline. So, you might not need an entire sub-area for it"*. He suggested moving encryption-related practices to the "secret exposure" sub-area, which we decided to do.

Several of the interviewees said that some practices were vague or broad, and on some of them they provided suggestions on how to make it more concrete and feasible. However, some of those practices were too hard for the experts to come up with a good solution for on the spot. Like interviewee 3 said about a specific practice within "secret exposure": *"It might be that it maybe shouldn't be such an open thing, but maybe get split up a bit in one way or another. But I don't have any good answer for what it would look like"*.

Within the "security testing" focus area, interviewee 4 suggested adding a sub-area for remediation of findings from the tests/scans. As he said: *"That could be a subsection of itself, remediation of those findings. It's a whole different practice. And involves implementing processes, as a company, or a team"*. He further elaborated on which practices could be part of that subsection:

> *I think the maturity model would just say: define a process. You know, it doesn't have to define the process for them, it would just state: you need to define a process. And that can be high-level like that, that's completely fine. And then some risk acceptance process should also be in there.*

Thus, a sub-area for remediation of findings, with practices for defining a process for remediation and defining a process for risk acceptance was added to the new draft.

Some of the practices in the initial draft were aimed at situations where external contributors were involved. Interviewee 5 meant that these situations in most cases are not relevant to the potential users of the maturity model. He said:

> *When you have a public repository that external contributors are pushing to, it's almost 99% for sure that this is not gonna be a company or a service that is really looking for a maturity model, if you know what I mean. This is probably gonna be some sort of open-source project that many people are contributing to. [...] But those kind of people are very, very probably not gonna also be doing a maturity model like this for their CI/CD, so I would almost remove the public repository stuff, and the external contributor stuff.*

We chose to remove the practice he referred to in this feedback, as we agreed that open-source projects were less likely to start using the maturity model.

When the modifications were implemented, the new draft had 10 focus areas, 40 sub-areas, and 216 practices.

4.2.2 Iteration 2 – Validating the practices

The main focus of the second iteration was the practices and their validity, relevance, and placement into maturity levels. However, we were still asking the experts about the focus areas and sub-areas to make sure that any contrasting views about the FAs and SAs were identified. For this iteration, 4 interviews were conducted with one individual per interview in all but one interview where two individuals from the same organization were interviewed together. None of the five experts we interviewed for this iteration had been involved in the earlier stages of the project.

In this round of interviews, several of the experts highlighted overlapping practices within different areas. As a result, we made an effort to reduce the overlaps by modifying or removing a selection of practices. Interviewee 6 suggested having a separate focus area for threat modeling, which at the time was placed within "orphan areas". Thus, we added a new focus area for threat modeling.

The experts we interviewed in interviews 8 and 9 were wondering why "certificates" was a sub-area for itself, with no practices in it. They said that certificates are only a kind of secret, and that the practices in the other sub-areas within the focus area of secrets also apply to certificates. Due to the fact that we did not have any unique practices to add to the sub-area of certificates, which we were recommended to add in the preceding iteration, we decided to remove it.

One important aspect of blocking deployment or merging was brought up by both interviewees 7 and 8. They said that blocking may not be beneficial in all situations. Interviewee 7 illustrated it well by sketching up this scenario:

> *Imagine that Vipps gets a huge problem. Nobody in Norway can "Vippse" money [make mobile payments to people or organizations]. A highly critical vulnerability shows up and must be patched before the patch fixing the payment problems can be deployed. It may cost millions for a business to be blocked from deploying the patch. That very thing is extremely dangerous. So, I would add a sentence with the possibility to override, because it is necessary*.

As she suggested, we added such a sentence for practices which involved blocking deployment or merging.

The "orchestration" sub-area was an area which 4 of 5 interviewees in the second round of interviews considered as out of scope for a CI/CD security maturity model. Interviewee 6 considered this sub-area as "*more like an operational thing*" and "*not necessarily a part of container security in this context*". Similarly, interviewee 7 said that "*if this is Kubernetes security [...] this is a completely different scope. A big, big thing which probably is a lot to cover in a master thesis about CI/CD*". Because of this feedback, we asked the two interviewees in interview 9 directly whether or not they considered orchestration as relevant for CI/CD security. They said that "*this is more general operation and Kubernetes-stuff*". However, they mentioned that it could be relevant if it was turned towards the build and configuration of clusters with CI/CD and Infrastructure as Code (IaC). Thus, we renamed the sub-area to "Orchestration configuration" and removed most of the original practices within the orchestration sub-area. These were replaced by practices for IaC-configured container orchestration, such as scanning IaC-scripts and Kubernetes manifests for insecure configurations. All of the new practices for orchestration were derived from feedback from the interviewees in the interviews of this iteration.

Another area which was considered as out of scope by some of the experts was the focus area of "skills and awareness". Both interviewees 6 and 7 suggested either making changes to the focus area or removing it. We ended up with removing all the practices within the focus area, to narrow down the scope by rebuilding it with only one practice for training team members on CI/CD and CI/CD security.

The sub-area for code signing in the "Artifact security" focus area got quite similar feedback from the interviewees (Table 23).

Interviewee 6	*I have never seen code signing. [...] I wouldn't say that these activities, for example, that you are listing in this first level are something that I have ever seen used in practice.* […] *I think that artifact integrity validation and then code signing part – that they are partially overlapping.*
Interviewee 7	*I have never seen anyone sign artifacts, and then we [the interviewee and her husband] talked a bit about it and came to the conclusion that it is a small market around it. [...] It cannot be in level 1 and we are actually thinking level 3, because it is such a small market around it that it will be difficult to set it up*.
Interviewee 8	*That by itself I would say is usually at least intermediate, because I have not seen a lot of signing implemented in different teams in different organizations. It's something that people aspire to do, so they want to do it, but they rarely actually get to it.* […] *Yeah, I think that it belongs to the code signing part as well. So potentially do merge them*. [«code signing» and «artifact imtegrity validation»]

Table 23: Feedback on the practices of code signing

Taking their feedback into account, we made some modifications to the matrix. The sub-area "Code signing" was merged into the sub-area "Artifact integrity validation", and some of the signing practices were moved from level 1 to level 3.

During some of the reviews in the interviews, we were made aware of some redundancy of container practices in different sub-areas:

Sub-area	Informant	Feedback
Container configurations	Interviewee 6	*And then there's container configurations – it has many of the same topics that were discussed earlier, rootless and so on. So minimizing the configuration of the containers, and I think that's a topic for container security that you had there earlier.*
	Interviewee 8	*"Images are configured to run as non-privileged users". I think we mentioned that in the container area as well that they should not run as root.*
Container (IAM)	Interviewee 6	*And then, again, there's "container run as non-root user", I think that's the third time.*
	Interviewee 8	*Again, under container, we mentioned least privilege and non-root, which is almost the same thing.*

Table 24: Feedback on redundant container practices

We realized that the container practices that were placed in other areas than the "Container security" focus area did not add anything distinctive, when comparing them. It also made more sense to have all container practices centralized in the focus area for "Container security". Thus, we removed the sub-areas "Container configurations" and "Container" from the focus areas "Configuration management" and "Identity and access management", respectively.

Interviewee 6 pointed out that "*misconfiguration, I think that's just flip-side of the configuration*" when looking at the misconfiguration sub-area in the focus area for configuration management. In addition to this, interviewee 8 said that one of the practices in this sub-area was heavily related to a practice in the sub-area for pipeline configurations. We took a closer look at all practices in the sub-area and concluded that all of the practices were already covered in other sub-areas. Thus, the sub-area "Misconfigurations" was removed from the matrix.

To make it easier to refer to the practices, interviewee 8 suggested giving each practice a unique ID:

> *When it comes to these practices maybe you want to give them some kind of short code abbreviations. Then you could maybe refer to them… say that this one is related to that, or something like that. For example pipeline security practice number one, so, PS001 or whatever, you know, just to have some kind of reference instead of referring to the entire paragraph*

Using IDs for the practices was implemented, since we saw the benefits of it both for communication and for the calculation of maturity.

We got even less feedback about moving practices to level four or five in the second round of interviews. Since only a few practices were considered so advanced that they could be put at a level higher than three, we decided to keep the structure with three levels. Levels 4 and 5 would be very empty, which also would mean that the difference between the highest levels only would be a few practices.

The resulting draft of this iteration consisted of 11 focus areas, 35 sub-areas, and 186 practices.

4.2.3 Iteration 3 – Designing the self-assessment solution and maturity visualization

For the third iteration, the focus shifted towards establishing an assessment solution for the maturity model. Based on the draft from the second iteration, we made control questions reflecting the practices. Simultaneously, another round of collecting feedback was conducted. This iteration was different than the previous ones, since we only did one interview. However, we also sent the draft from the second iteration to all 10 participants which were interviewed in iteration 1 and 2. Additionally, it was sent to one of the informants we did a pilot interview with a few months before the start of the master project. The draft was sent by e-mail, with a one-week deadline for them to send written feedback. Only 4 out of 11 informants had time to provide feedback, and two of them wrote, respectively, that "*it looks very good*", and "*I like the revisions. [...] It looks great and in my opinion is directly applicable and practical*". The two other informants gave some comments with suggested modifications that could be implemented to improve the model.

Interviewee 8 wrote that one of the practices should be distinguished more from another practice: "*S_LC_04: I would elaborate on this point to distinguish it from S_SM_05, since expiration of 1 year also makes credentials temporary. Perhaps it could be reworded to something like 'Utilise short-lived tokens for service integration'*". We decided to replace the word "temporary" with "short-lived or dynamic".

Interviewee 11 suggested adding managed identities as a practice on the highest maturity level within the sub-area "secret management":

> *Maybe it is something to add to level 3, managed identities, something that can replace management of secrets almost completely. […] From my experience, I would need this in level 3 to define maturity in my group. I would say that if you implement managed identities, it means that the secrets that the identity needs access to are solved with a least privilege, zero trust, RBAC way of thinking. Then you limit the access to the exact identity, and that identity is managed. It can be a traditional service account, but it can also be something a little bit different. Many of the things we talk about now, in many of the other teams working where I work now, is 'how to go away from secret rotation and secret expiration until it's just managed for you'. Your application uses a managed identity, your application is able to get hold of what it is supposed to because the access is controlled on an identity level without involving secrets, which also applies to pipeline security – to a certain extent.*

This practice would collide with the practices which already were a part of the sub-area, since it almost completely replaces management of secrets. It would be impossible to achieve the highest level if we incorporated it into the existing levels, since the answer would be no for the compliance with the other practices. Thus, we placed it in a new column we called "stretch goals", intended for "bonus practices" that would be even more favorable. This is an idea that would be interesting to fully incorporate into the model, but due to time constraints we could not prioritize further development with this new aspect.

For the practice of using key vaults for secrets, interviewee 8 wrote this feedback: "*S_SM_02: some secrets may be stored directly within the system using them, e.g. as masked variables in GitLab or secrets in GitHub. Setting up a vault may be not trivial. May want to move this to Level 2*". This practice was originally placed in level 2 in the initial draft based on the literature reviews. However, in the first round of interviews, interviewee 5 shared his view on the placement of the practice:

> *I can see the reason for having the secure key vault stuff as a level 2, but what I do think that in this time, this should - like in this day and age – it should almost, ALMOST, be a level 1, it's like a 1.5. I think anyone who is not doing this already… is setting themselves up for failure sooner or later. Because if you don't have your secrets in some sort of key vault, where do you have them then? That's my next question. You need to be storing those secrets somewhere, so are they just in, like, a file that everyone has access to or something? So, to me – that would be a huge*

security fault. So, I would almost move the "Keys are stored in a secure key vault" to level 1.

Based on this specific comment, the practice was moved down to level 1 in the draft reviewed by interviewee 8. It may be that interviewee 5 and interviewee 8 perceived the term key vault differently. We interpreted both experts to mean that secrets should be stored securely in well-fitting systems or platforms, and that this can be considered basic. Instead of moving the practice to level 2, we reformulated it to also include storage in the CI/CD platforms. In this way, secure storage of secrets was kept in the level of first priority practices within the focus area for secrets.

Informant 12 meant that "*the principle of least privilege is quite more advanced than level 1 in my experience*". As a result, we decided to move the practice regarding the use of least privileges for secrets to level 2.

The draft which was reviewed in the third iteration had a practice for having a policy to prevent secrets from being committed to the code repository or printed to logs and consoles. Interviewee 8 wrote this about the practice: "*S_SE_01: regarding exposure of secrets in logs, I'd reword it to something like 'Mask plaintext secret values in logs' – it should be implemented, and not just a policy. Potentially split this from the part about committing secrets*". Based on this feedback, we split the practice into three separate practices. We kept the practice for having a policy, while we also added the suggested practice of masking secrets in logs. The third practice which was a result of the split was added to keep the original perspective which also included console outputs of builds. For this practice we took a new look at the pool of practices identified from the knowledge base and formulated the practice to reflect the relevant practices found in the knowledge base.

There was another policy-practice in the "Secret exposure" sub-area which interviewee 8 left a comment on: "*S_SE_02: I personally don't recall seeing such verification in practice – perhaps move to Level 2?*". After a new review of the practices identified from the knowledge base, we modified the practice from requiring a policy to requiring having an established process for verification of removal of secrets in artifacts. We did also follow his suggestion to move the practice from level 1 to level 2.

For the practice aimed towards scanning of containers running in clusters, interviewee 8 pointed out: "*containers don't have to run in clusters 😊 Perhaps 'Scan containers in live environments for vulnerabilities'?*" We reformulated it to the exact formulation which was suggested. He also wanted a practice to be more specific: "*'security best practices' sounds rather vague – could this refer to any specific practices perhaps, e.g. from OWASP?*" To make it less vague, we added three examples from OWASP, CNCF, and NIST.

Integration of security checks into pre-commit hooks was a practice that interviewee 8 seemed to be skeptical to:

> *I've never seen this implemented, and suspect devs would reluctant* [sic] *to do it as a pre-commit hook – if the code is scanned by the pipeline before deploying (as it should), then there is usually little harm in committing vulnerable code to a feature branch. Since such scans can take a while, devs may want to run it manually when they want to check their code before committing; but doing it before every commit sounds excessive.*

Thus, we removed this practice.

Additionally, interviewee 8 had some comments on minor reformulations and overlapping practices, which were taken care of in the new draft.

Informant 12 suggested some additional practices: "*1 pipeline per environment, and any other deploy can be seen as security incidents (than from the correct pipeline)*" and "*SCA checks whether the vulnerable part of the library is possible to hit, AKA actually exploitable lvl3*". He also suggested a third practice: "*build systems are rebuilt nightly with new patches applied to base image*". All of the suggested practices were added to the model.

In the previous iteration, the focus area "skills and awareness" was trimmed down instead of removing it completely. However, interviewee 11 was a little doubtful about the relevance of the focus area:

> *I don't know if «Skills and awareness» actually belongs here […]. If you work in a team where Security Champions are defined, I would maybe have thought that they owned this creation of security awareness, and maybe would be part of something internally which provides training for developers. Maybe they manage it, maybe they bring in something external? Maybe you use an external actor for such things. So, the spot for «skills and awareness» inside the maturity model was a bit… Yeah, I would maybe have kept it outside.*

Since three interviewees (6, 7, and 11) commented on the relevance and scope of the focus area, we decided to remove it from the model. We tried to scope it to only CI/CD security skills and awareness in the previous iteration, but with only one practice it did not make sense to dedicate a whole focus area for it. However, addressing skills and awareness within CI/CD security is still important, and we would like the model to incorporate it in some way. In this third iteration, the focus area was removed without finding a new placement for the single practice it consisted of.

Interviewee 11 suggested renaming a sub-area in the focus area "Identity and access management": "*The sub-area named 'Lifecycle of identities' – I thought that 'Identity management' can be another name for it. Just because, in my head, identity management was*

something that can apply to a little bit more as well in that sub-area". He followed up this with suggesting a new practice:

> *Identity federation, typically between clouds is something that I would have noted here. Because if you think of a CI/CD setting where you have pulled in something from another cloud; If you work with GitHub Actions and work with resources in Azure, and then you go get something from Google's GCP. Then it is possible that such a federated identity is smart to consider, to not have too many service accounts and privileged accounts in play. So, I would maybe add federation of identities. This kind of becomes level 3.*

We implemented his suggestions by renaming the sub-area to "Identity management", and adding a practice for identity federation at level 3 in this sub-area.

We had a practice for enforcing denial of force pushes to branches, but interviewee 11 had a comment on this:

> *Force pushing – we have recently talked about this quite a lot where I sit now. I think that force pushing is completely fine as long as it's not on "main". As long as it's not on the branch going out to production, force pushing is OK. Because it's about different developers having different flows. Some want to force push on their PR-branch before it is ready for PR, and if you want to do that, I won't bother you. As long as I get an overview of what you have done, I'm not too concerned about preventing it. Sometimes, such well-intentioned advice hits a bit hard on the toes of developers who work in different ways. So, learned from own mistakes we have softened it up a bit.*

The practice was softened up a bit in our model as well, to only deny force pushing to persistent branches.

Within the first sub-area of the focus area "Monitoring", interviewee 11 shared his thoughts:

> *I was wondering a bit about "Pipeline predictability". Do you think of reproducible and deterministic builds that I can run on my own machine? Because, in that case, I would add – at the first level of maturity – that the tools used in the pipeline can have the same outcome and result if I run on my own machine or if I run in a hosted CI/CD pipeline. Just because my team and my last couple of years have been very focused in the direction of "reproducible everything". I must be able to run on my machine, everything that also is run out in the CI/CD of tests and builds. And the outcome, given that we have the same configuration of tools and platforms should be completely, completely the same. And that's very important for the security part as well, that you have scanners that catches the same locally like if they run in a pipeline. Because then you can start like you say here, to have meanings about tagging, the artifacts, SBOM, signature, and everything becomes completely the same*

> *if you have a tool suite which works locally and in the pipeline. It could be maturity level one, in "pipeline predictability" that it is reproducible on developer machines as well. At least that a developer can get feedback from local machine.*

The basic level in «Pipeline predictability" was empty at that time, so this suggestion helped us with consummating the sub-area as complete with practices on each level.

Already at the start of the iteration, we started formulating the control questions. Even though we had not received all of the feedback on the latest draft yet, we started creating the questionnaire with questions based on the draft we sent to the informants. However, any of the modifications made to the matrix during this iteration were also implemented to the self-assessment questionnaire by modifying the control questions to reflect the updated model.

When the control questions were formulated, we started to work on the calculation and visualization of the results based on the answers in the questionnaire. Thanks to the self-assessment tool developed by Tokerud and Jansen (2022), we had a reference to look at. Much of the calculations in our Excel sheet are influenced by the solution developed by Tokerud and Jansen (2022). However, we solved some aspects differently. Since our self-assessment questionnaire was structured differently than the one in the reference model, the retrieval of the answers had to be done in another way. Our questionnaire did not have the name of the maturity level in each row for each practice. Thus, we solved it by adding a hidden column with a maturity ID, to be able to differentiate between the practices based on which level they belong to. Similarly, we did not have the names of the focus area and sub-area in each row for each practice. Since we had IDs for the practices, with a prefix reflecting the focus area and sub-area, the prefixes of the IDs were used to differentiate which areas the practices belonged to.

Two of the questions in the questionnaire were constructed in a way that the answer would affect two maturity levels. This led to three different outcomes. Either, the answer would be "no" which would mean no points to any maturity level, or the answer would be "yes, manually" which would mean one point to the intermediate level. The third option, "yes, automated", would mean one point to both the intermediate level and the higher level. Because of these two questions, the calculations had to be done differently for these two specific cases.

Another difference when comparing our calculation and visualization of maturity with Tokerud and Jansen (2022), is that we made the visualization a bit richer of information. We added calculations and visualizations of how many of the practices that have been implemented per sub-area, and a percentage for the progression on each maturity level within the distinct focus areas. Additionally, a percentage showing the proportion of implemented practices within each focus area was implemented in the maturity dashboard.

The third iteration resulted in a draft consisting of 9 focus areas, 32 sub-areas, 175 practices, and 173 control questions.

4.2.4 Iteration 4 – Case study evaluation

At the end of the project, a naturalistic case study evaluation was conducted. Two participants from a large Norwegian organization in the finance sector were invited to an online session (on Microsoft Teams) where they filled out the self-assessment questionnaire, while we, the researchers, observed and were available for potential questions. After the questionnaire was completed, the participants got to see the results in the dashboard-sheet. We then conducted an interview with the participants to get more explicit answers to whether or not the model met the predefined evaluation goals.

During the session where the participants filled out the self-assessment questionnaire, there were some of the control questions where they had comments or wanted clarification on. As an example, one of the control questions asked if container base images and container runtime are kept up-to-date, and informant 13 responded: "*depends on what up-to-date is, because if it's like a, for example, if it's new features that we don't need, we sometimes might not update. But security patches are always included*". This comment and other feedback we received was addressed through revisions of control questions.

The version of the model which was used in the evaluation had percentages for how many of the practices within each focus area were implemented, and percentages for each maturity level in the respective focus area of how many practices that were implemented. However, the only measure which was provided for the overall result was the overall maturity level. Interviewee 8 said this about getting 0 as their overall maturity level, when many of the sub-areas were scored at levels 1, 2, or 3:

> *Looking at the lowest score, is a bit rough, so maybe try to show that, you know, we're not doing that bad. It's just this particular item that's a gap.* […] *I was just hoping for a higher score, that's all. But I think, if you look at the percentages then it's pretty good - pretty much what we expected.*

As a response to this feedback, we implemented an overall percentage as well. If this had been a part of the dashboard in the evaluation session, they would have seen that they were compliant with 83% of all practices in the model. It sounds very strict to get 0 as maturity level, when more than four fifths of all practices in the model are implemented. However, the levels were intended to comprise a prioritization order, with level 1 consisting of practices that can be considered basic and should be prioritized when beginning working with CI/CD security. Thus, in the aftermath of the evaluation session, we decided to take a closer look at the basic practices where the participants in the evaluation answered "no". The purpose was to assess the question formulations and the level placement, to consider whether any questions should be reformulated or moved to another level. For the questions where they answered "no", we decided to keep their formulations and level placements. However, 21 other questions were modified, either by reformulating them or replacing them to other levels or sub-areas. These changes were based on the evaluation participants' comments when they filled out the questionnaire.

In the interview after the evaluation session, the participants had a lot of positive feedback, in addition to pointing out a few points for improvement:

Positive feedback	Points for improvement
It's good to visualize it, and now we see where we have the biggest gaps.	It would also be good to use this when it comes to comparison to other organizations. So then we can see how much we are relatively to the other similar organizations of similar size, similar industry, because the expectations will be different.
I'm quite happy with this exercise that has highlighted a few things for us as well.	Duplicated or very similar questions
This is most useful when you want to advocate for security, especially with non-tech people. They love seeing this.	Overlapping questions
I think it can help our organization, and a lot of others, identify what things are lacking. And especially highlighting the areas where there are big gaps, you know, so not just individual things, but it's good to aggregate it and show like, you know, we are… pipeline security for example is not doing that great, or you are really lacking in supply chain and third-party risk assessments. So I think from that perspective it's good because you might know that you have these 20 different gaps, but then defining that, you know, in this particular area you have biggest weaknesses. That's one of the benefits in it.	There was a lot of focus on SBOM, but as [name of informant 13] mentioned, SBOM is not widely adopted in the industry yet, so people do produce it, but they don't know what to do with this. So maybe reduce or consolidate some of the SBOM questions so it doesn't have such a big impact on the score.
It can definitely show the most glaring pain points.	Some questions were maybe a bit confusing to understand.
I think it covers a lot of different aspects when it comes to pipelines.	There was one question that didn't seem security related
It's quite comprehensive, I'd say.	Another thing was the context of the questions, because a lot of… there are some of the questions where we said "umm, uh, it depends on the context and the criticality" and stuff like that. And, we mentioned criticality a lot of times and criticality comes from risk assessment. So maybe that's something you would want to ask about as well, like risk assessment in general, because that's when companies determine how critical a pipeline or an app is, and depending on that, the answers for a lot of the questions here will be very different. So maybe you could also even specify the questions and say for critical systems or for non-critical systems as well, because there is always a threshold where things get quite serious when it comes to rigorous implementation of security and things where it's a little bit more loose because there isn't much to lose.
I'd say this one is quite unique, because the ones that I work with tend to be generic and they don't exactly focus on the pipeline. Even though the pipeline has	I also have a feel that it could be easier to use. I don't know how. Like, if you ask me about any suggestions to improve it, I don't have any. But I feel

shown to be, uh, quite lucrative attack surface and the dev environment in general. So in that sense, I would say it is kind of novel in that sense.	like it could be easier. Maybe if it were, say, a web page, right, just click things and then it generates the summary. Maybe that's… Because there you have much more flexibility, right? And you could make it… You could structure it page by page instead of just one long page to scroll.
So here we definitely can see the focus is on the pipelines, even though a lot of other additional things are taken into account like configuration management like identity and access and monitoring. So those are all… ancillary aspects. Additional perspectives, which are still of course related to pipelines, so I think it covers the pipeline security from multiple dimensions, which is really good.	I mean, the dream would be to have a dashboard and then some process that would automatically measure all – like, most of these questions automatically. And then just reports back to the dashboard and we can just take a look there and see: "Ohh, this month we reach this level." […] But yeah, that's a good start, I'd say.
And I think this will be more and more relevant. Now that platform engineering is becoming its own thing, because platform engineering tends to rely on pipelines a lot. And this is something that would help. To kind of measure the maturity of those kind of platform engineering pipelines.	
I'm glad we did this exercise.	
It was quite fun actually.	
Yeah, this model could become a tool for security engineers to highlight and emphasize the gaps that their teams have.	
I like it. It's pretty well structured, I'd say.	
Looks like you have put quite a lot of effort into this, with good questions and good well conducted interview as well. In this case study with us. Think you guys are doing a good job and yeah, I definitely see that this could be in fact useful in the industry. You know, it's not just a little project that you complete and then nobody uses it.	
It's actually something that you could take forward and then if you end up working in some security consultancy or some organization, then you could bring this to the table with you.	
And academically, you could also pursue to publish this as well. I think it will gain attraction.	

Table 25: Evaluation feedback - Positive aspects and points for improvement

After the last changes based on the evaluation feedback were implemented, the final version of the maturity model consisted of 9 focus areas and 31 sub-areas. It ended up with a total of 173 security practices that are assessed through 171 control questions in the self-assessment questionnaire.

4.3 The final artifact – The CI/CD Security Maturity Model

Through an iterative design process and a case study evaluation, the project resulted in the first released version of the CI/CD Security Maturity Model. The artifact is intended to facilitate evaluation of CI/CD security maturity and make the maturity status comprehensible,

not only for technical roles, but also for managerial roles. It gives an objective score and highlights which areas there is room for improvement.

The core elements of the artifact are the self-assessment questionnaire and the maturity dashboard provided in an Excel spreadsheet. When the control questions in the self-assessment questionnaire are answered, the maturity is calculated under the hood. The maturity dashboard will always visualize results based on what is filled out in the questionnaire. A change made to an answer will immediately be incorporated into the calculation sheet, which the dashboard visualization retrieves the data from. In that sense, the dashboard is dynamic by constantly updating its visualization based on the questionnaire answers. This makes it a good tool for tracking the progression, where the current status of the maturity always is provided (as long as the answers are updated to reflect the current practice).
The spreadsheet constitutes a good foundation for working with CI/CD security, both for upper management, security managers, and technical professionals. It provides an overview of maturity which is understandable no matter the level of technical expertise of the user. The people with insights into the technical details required to answer the control questions can fill out the self-assessment questionnaire, and the results can be presented to the security management. The security management can then use the results to argue for investments in CI/CD security improvement initiatives to the upper management.

Based on multivocal literature reviews and interactions with domain experts, a total of nine focus areas were included in the CI/CD Security Maturity Model:

1. Secrets
2. Container security
3. Security testing
4. Artifact security
5. Pipeline security
6. Software supply chain and third party risk
7. Configuration management
8. Identity and access management
9. Monitoring

When filling out the self-assessment questionnaire (Figure 6), the users can choose between “No”, “Partial”, “Yes” or “N/A” as their answer to each control question. Answering “N/A” will exclude the corresponding practice from the maturity calculation. This means that if the users deem the practice as “not applicable” for their context, they avoid getting “punished” for not implementing it. “Partial” is an option which gives half of the full score for the practice, and awards that the organization has taken steps to get closer to achieving the requirements of the practice. In the rightmost column, it is possible to write a comment if an explanation for the answer is considered relevant or useful. There is also a column called “options” in the questionnaire. This column is only used for two particular control questions, where “yes” or “no” is not sufficient to decide the maturity level. In these two cases, the users

also have to choose either “manual” or “automated” if they answer “yes”. The “manual” option is tied to the midst maturity level, while “automated” indicate the highest level.

Focus Area	Sub-area	ID	Maturity Level	Control Question	No	Partial	Yes	N/A	Options	Answer	Comment
Secrets	Lifecycle of credentials	S_LC_01	Basic	Are all default values of credentials related to CI/CD processes changed?			x			Yes	
		S_LC_02	Intermediate	Does your organization/team have mechanisms to detect stale credentials related to CI/CD processes?			x			Yes	
		S_LC_04	High	Does your organization/team utilize short-lived/dynamic credentials for CI/CD processes?			x			Yes	
		S_LC_03,05	Int. / High	Has your organization/team implemented automatic detection of leaked secrets? Manual or automated rotation?	x					No	
	Secret management	S_SM_01	Basic	Are secrets stored in a secure key vault (includes CI/CD platform's secrets storage)?		x				Partial	
		S_SM_02	Intermediate	Is the principle of least privilege applied for access to secrets and privileges of secrets?		x				Partial	
		S_SM_04		Does your organization/team use separate keys for separate environments, tasks, and roles?		x				Partial	
		S_SM_05		Is a policy for using expiration dates of secrets, limiting the secret validity to a maximum length, defined?		x				Partial	
		S_SM_03,06	Int. / High	Does your organization/team have routines for rotating and revoking secrets? Manual or automated?			x		Automated	Yes	
		S_SM_07	High	Is retrieval of secrets within CI/CD processes done dynamically and passwordless?			x			Yes	
		S_SM_08	High	Is a policy for using expiration dates of secrets, limiting the secret validity to a maximum length, enforced?			x			Yes	
	Secret exposure	S_SE_01	Basic	Do you mask plaintext secret values in logs?			x			Yes	
		S_SE_02		Does your organization/team use vendor options or third party tools to prevent secrets from being printed to console outputs of builds?			x			Yes	
		S_SE_03		Has a policy for preventing commits of plaintext secrets to the version control system been defined?		x				Partial	
		S_SE_04		Does your organization/team automatically scan code in CI pipelines to prevent commits of plaintext secrets to the version control system?	x					No	
		S_SE_05	Intermediate	Does your organization/team use server-side pre-receive hooks to prevent commits of plaintext secrets to the version control system?			x			Yes	
		S_SE_06		Has your organization/team established a process for verifying that secrets are removed from artifacts such as container images, binaries, and Helm charts?			x			Yes	
		S_SE_07	High	Does your organization/team use client-side pre-commit hooks to prevent commits of plaintext secrets to the version control system?		x				Partial	
		S_SE_08		Are secrets encrypted with customer managed keys (CMK) at rest?			x			Yes	
Container security	Container image scanning	CS_IS_01	Basic	Are base images systematically scanned for vulnerabilities before any additional development is performed on them?			x			Yes	
		CS_IS_02		Are issues identified in scan results remediated on an ad hoc basis as they arise?			x			Yes	
		CS_IS_03	Intermediate	Are containers running in live environments regularly scanned for vulnerabilities?		x				Partial	
		CS_IS_04		Do you have established routines for taking action based on the results from image scanning?			x			Yes	
		CS_IS_05		Are there mechanisms in place to prevent the deployment of containers with detected vulnerabilities, while allowing for overrides under certain conditions?			x			Yes	
		CS_IS_06	High	Is there continuous monitoring for embedded malware in your environment, using both malware signatures and behavioral detection heuristics?			x			Yes	
	Orchestration configuration	CS_OC_01	Basic	Is the build and configuration of container orchestration conducted using Infrastructure as Code (IaC) and CI/CD processes?			x			Yes	
		CS_OC_02		Do the declarative configurations of container orchestration adhere to security best practices? (e.g. OWASP Kubernetes Security Cheat Sheet or CNCF Cloud Native Security Whitepaper or NIST SP 800-190)		x				Partial	
		CS_OC_03	Intermediate	Are IaC configurations and Kubernetes manifests scanned for insecure configurations (risks) before deployment?	x					No	
		CS_OC_04	High	Is container security runtime, including Security Contest and container capabilities, taken into account for your projects?	x					No	
		CS_OC_05		Are containers continuously scanned for vulnerabilities and misconfigurations while running in a cluster?	x					No	
	Maintenance	CS_M_01	Basic	Are your base images and container runtime kept up-to-date?		x				Partial	
		CS_M_02	Intermediate	Do you schedule pipelines to regularly update base images and test them afterward to assure their functional ability?			x			Yes	
	Privileges	CS_P_01	Basic	Are containers executed in rootless mode in your environment?			x			Yes	
		CS_P_02	High	Are security profiles implemented for containers?			x			Yes	
	Origin verification	CS_OV_01	Basic	Do you ensure that images are downloaded only from trusted and reliable sources?			x			Yes	
		CS_OV_02	Intermediate	Do you only allow images and registries that are trusted within your CI/CD processes?						Blank	
		CS_OV_03		Do you use a private registry and download external container images via a proxy to enhance security?						Blank	
		CS_OV_04	High	Do you verify or validate images by cryptographic signatures?						Blank	

Figure 6: The self-assessment questionnaire

The model consists of three maturity levels, “basic”, “intermediate”, and “high”. These levels are intended to represent a progression of maturity, from essential capabilities to more advanced capabilities within CI/CD security. The “basic” level is composed of CI/CD security practices that should be prioritized before the practices in the higher levels. Another characteristic of the level 1 security practices is that they will have a good effect on the CI/CD security, without requiring tremendous resources. The “intermediate” level is supposed to be a natural continuation of level 1, by adding security practices that will improve the maturity, but not intended to be prioritized initially. Then, the “high” level builds upon the preceding levels and consists of security practices that are a bit more advanced, but important to be able to secure the organization against a broad range of threats.

The maturity dashboard (Figure 7) provides results in three different aggregation levels. The lowest aggregation level is the sub-areas. For each sub-area, there are three metrics which are presented: (1) percentages for the maturity levels (the proportion of implemented practices from the maturity level within the sub-area), (2) maturity level of the sub-area, and (3) how many of the practices from the sub-area that are implemented. The middle aggregation level shows the focus areas and presents two metrics: (1) the maturity level of each focus area and (2) a percentage of how many practices within the focus area are implemented. At the highest aggregation level, the overall maturity level is presented along with a percentage of how many of all the practices are implemented.

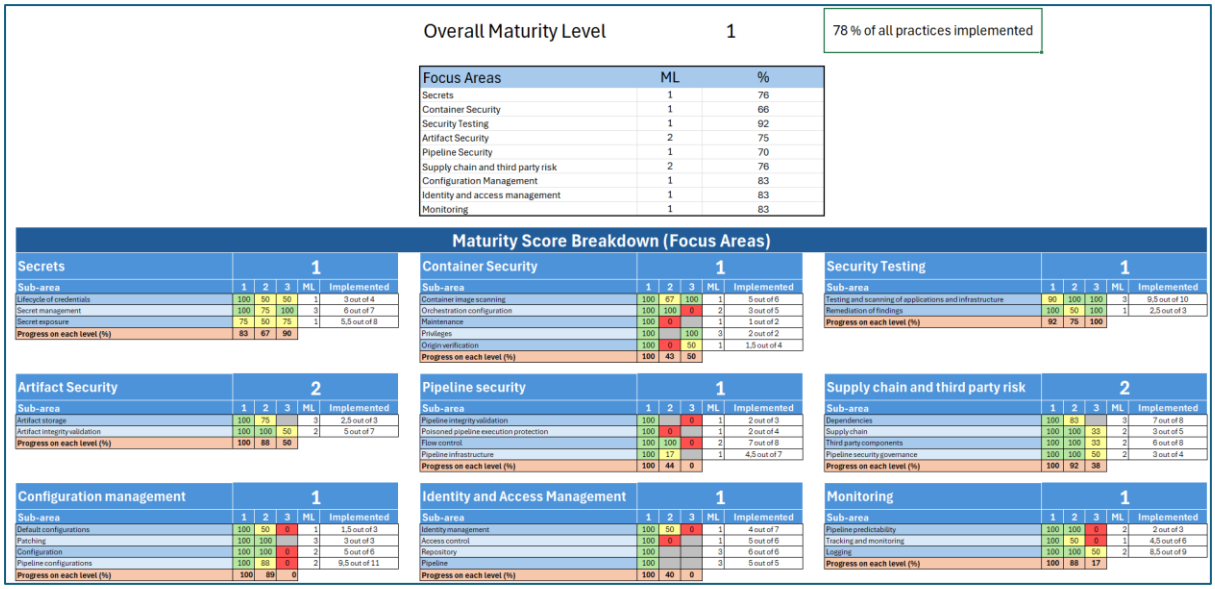

Figure 7: The maturity dashboard

To achieve a maturity level within a sub-area, at least 75 percent of the practices in that level (and the preceding levels) must be implemented. This means that maturity level 3 will only be achieved if all maturity levels have 75 percent or more compliance within the sub-area. The maturity level for the focus areas reflects the lowest level achieved among the sub-areas within the focus area. If one or more sub-areas get 0 as its maturity level, the whole focus area will get 0 as its level. The same applies to the overall maturity level, as it reflects the lowest level achieved among the focus areas. The reason is that the model is designed to encourage fulfilling the lower levels before prioritizing the practices in higher levels. There is a reason why the practices are placed in their respective maturity levels, which therefore is reflected in the scoring system of the maturity model.

Due to the additional "options" field, which is used for two of the control questions, the calculation of maturity had to be tailored to reflect the exception from the rest of the sub-areas. This only affects the two first sub-areas in the model. For those areas, a special maturity level is assigned to the special questions. They are marked with intermediate/high, since the question covers a practice which spans over two levels depending on the answer. Thus, the scoring for maturity levels "intermediate" and "high" is done slightly differently for the two particular sub-areas.
For the midst maturity level, we count the number of answers with "yes" and "partial" for the questions at the "intermediate" level and the number of answers with "yes" combined with "manual" and "yes" combined with "automated" at the "intermediate/high" level. "Automated" is counted to make sure that you are not "punished" for being compliant with the highest level, when you have replaced the manual practice tied to the "intermediate" level with an automated practice. Without counting "automated", you would lose a point for the midst level when you comply with the automated practice. For the highest maturity level, only the combination of "yes" and "automated" for the "intermediate/high" level is counted in addition to "yes" and "partial" on the "high" level.

Some of the sub-areas in the maturity model do not have practices on certain maturity levels. For the sub-areas with practices on only one or two maturity levels, the highest maturity level can be achieved when 75 percent of the practices in the maturity level(s) with practices are implemented. As an example, the “Repository” sub-area in the focus area for “Identity and access management” only has practices in the lowest maturity level (basic). As long as at least 75 percent of the practices are implemented, level 3 is achieved. This way, the empty maturity levels do not hinder the possibility for achieving the highest level. At the same time, these sub-areas still keep their significance in the model, and cannot be ignored if you want to achieve level 1 or higher levels as the overall maturity level.

The first release of the maturity model is made available here.

5 Discussion

5.1 Reflections on the design of the maturity model

Reviewing the outcome of the CI/CD Security Maturity Model, it becomes clear that adopting design science research principles to create a functional model or artifact has proven beneficial. The process began with two multivocal literature reviews: the first to identify focus areas, and the second to pinpoint relevant security practices. These reviews enabled the drafting of a model grounded entirely in scholarly and industry literature.
Subsequently, the model underwent three iterative cycles, each involving feedback and data collection from a diverse group of industry experts. This iterative approach helped refine and strengthen the model, incorporating new insight based on expert feedback.
The CICDSecMM facilitates an organization's evaluation of its CI/CD pipeline security by using a structured questionnaire. The responses are quantified into numerical scores, which are then aggregated and analyzed across different sub-areas and focus areas to classify the organization's maturity into levels 0, 1, 2, or 3.

To address the first supporting research question (SRQ1), "What are the critical/essential focus areas that must be considered when designing a security-oriented maturity model for CI/CD pipelines?" we initially identified ten focus areas from the literature reviews. Following interviews with industry experts, we refined this list to nine, excluding "skills and awareness" after validation.

For the second supporting research question (SRQ2), "How can the security practices be effectively mapped to different maturity levels to reflect incremental security improvements?" we have identified 173 practices from our literature review and consultations with industry experts. These practices are distributed across three levels of maturity, ensuring a structured approach to enhancing security incrementally.

The output from these research questions resulted in a matrix that maps a wide range of security controls against various sub-areas within the identified focus areas, each graded according to a selected maturity level. This matrix serves as the underlying structure for both

the structured questionnaire and the dashboard visualization. These tools are designed to function based on the input from respondents, allowing for dynamic interaction with the underlying data.
Furthermore, this enables organizations to evaluate a wide range of security-related aspects within their CI/CD pipeline operations, providing them with an objective assessment of their security posture. This, in turn, highlights areas that are deficient and require improvements.

5.2 The unique qualities of the CI/CD Security Maturity Model

What is unique with the CI/CD Security Maturity Model is the comprehensive coverage of important aspects of CI/CD security. There are many maturity models which focus on secure development, but they tend to cover a wide range of aspects of software development. The wide focus compromises on the depth of the coverage, which is one of the reasons why the generic software development security maturity models are not sufficient to measure CI/CD security maturity.

By focusing the model exclusively on security and directing all efforts toward enhancing security within CI/CD pipelines, it emphasizes the importance of robust security measures. This model consolidates all security aspects under one framework, thereby offering thorough coverage and fostering a dedicated approach to security in CI/CD environments.

None of the security-focused maturity models we have identified during our work on this project mention pipelines, except for OWASP SAMM, which has two sentences with "pipeline" in the "Secure Build" practice of their model. Pipelines are essential parts of CI/CD, enabling automation of routines for testing, integration, build, deployment, and updates defined with declarative code. Thus, by having pipelines as a focal point in our maturity model, we contribute with a significant enhancement of the attention brought to CI/CD security in maturity models.

5.3 Assessing the artifact using the evaluation goals

Evaluation is crucial in design science research, and by using well-executed evaluation methods, the utility, efficacy, and quality of the designed artifact must be demonstrated (Hevner et al., 2004). Becker et al. (2009) also emphasize the importance of evaluation by listing it as one of the requirements for the development of maturity models. Thus, a set of evaluation goals were defined during the design phase, to be used for comparing the final version of the artifact up against (the evaluation goals can be found in Table 21).

To evaluate whether the model can be used by organizations or teams to self-assess the current CI/CD security maturity, we predominantly relied on the case study evaluation. Based on the naturalistic evaluation session and the interview after the participants were finished with the self-assessment, our general impression is that the participants thought the self-assessment overall was easy to use. However, there were some control questions they found harder to understand than others. They provided some comments and suggestions on several control questions, and most of them were revised to reflect their feedback.

Throughout this project, 13 experts were involved in the design and evaluation of the maturity model. When we have collected feedback, most of the informants have used words such as “relevant”, “applicable”, and “practical” about the model. Their contributions through their comments, reflections, and suggestions have further improved the applicability of the maturity model. Therefore, the evaluation goal stating that “the model is applicable for organizations with CI/CD pipelines” has not been disproved yet. However, to validate the fulfillment of this goal, more people and organizations must test and evaluate the maturity model.

Whether or not the model can be used to improve security capabilities within CI/CD for organizations or teams cannot be concluded on based on the activities conducted in this project. If the model actually can help with improving CI/CD security capabilities cannot be verified until there is conducted a case study which follows up the CI/CD security of a team/organization over a period of time when the maturity model is actively used. Due to the time constraints of the project, we were not able to conduct such a time-consuming evaluation of the artifact. However, the security practices in the model are placed into maturity levels with an intention to progressively improve the security maturity. The experts that have contributed to the project have reviewed the levels of the practices, and provided feedback when they identified practices that in their perception were misplaced, incorrectly worded, or irrelevant. Thus, one could state that the maturity model *in theory* can be used to improve CI/CD security capabilities, even though it is not practically proved.

Based on the naturalistic case study, we found that some control questions were ambiguous or formulated more complicated than necessary. This does not adhere to the evaluation goal of having comprehensible security practices, which made us revise these questions before releasing the model. Since these revisions were done after the evaluation, we cannot conclude whether the revisions were effective. When it comes to the relevance and placement of security practices, the experts involved in the design helped with assuring the relevance and correct placement. Our perception is that the practices are relevant and placed in appropriate levels, since we have utilized both the knowledge base and the environment (domain experts) when deciding practices and their placement.

Informants’ enthusiasm and desire to adopt the maturity model into their organization was found in most interactions we had with the experts. To what extent varied, which is to expect since people express themselves differently. Nevertheless, our perception of the informants’ impression of the model is that they see the utility and relevance of such a maturity model. This was explicitly mentioned by the informants participating in the case study evaluation.

The number of suggested modifications was lower for the draft used in the evaluation, when compared with the previous phases of the design iterations. This is as expected since the artifact was progressively polished and improved throughout the iterations. Another factor which may have influenced the decrease in suggestions is the lower number of informants involved in the last evaluation (the summative evaluation). The model was largely adjusted according to the results of the evaluation, which in theory should mean that a new round of evaluation with the same organization would result in very few or no new suggestions for

modifications.

5.4 Practical implications

The CI/CD Security Maturity Model (CICDSecMM) is designed to help organizations assess and enhance the security of their CI/CD pipelines. It utilizes a self-assessment spreadsheet that translates responses into a numerical system, which allows for the quantification and aggregation of maturity levels. This aggregated maturity level simplifies communication about CI/CD security, making it more accessible, especially to non-technical stakeholders. The more aggregated the maturity level, the more abstract and understandable the communication becomes.

The model offers several benefits:

1. It covers critical areas relevant to CI/CD security and facilitates easy identification of improvement opportunities through an intuitive dashboard.

2. It provides a snapshot of the current security posture, with straightforward follow-up activities and immediate visibility of improvements in the security score.

3. It simplifies reporting on security through the aggregation of maturity levels, from detailed sub-areas to broader focus areas and up to a high-level overview of the overall maturity score.

Organizations can use the CICDSecMM to evaluate and enhance their CI/CD security by completing a spreadsheet. The system automatically calculates the maturity level for each sub-area, aggregates these into each focus area, and then culminates in an overall maturity level. This process provides a clear snapshot of the current state, aiding in planning and taking concrete steps to improve security.

Moreover, the model supports detailed reporting by leveraging all levels of abstraction, from the most granular details in sub-areas up to the highest abstraction of the overall maturity level. This feature is particularly useful for tailoring communications to the recipient’s level of technical understanding.

5.5 Opportunities for further research

5.5.1 Expanding the focus areas (and sub-areas)

Although the CICDSecMM has been significantly developed, validated, and refined over multiple iterations with input from a wide range of industry experts, there is always room for improvement to adapt to emerging trends and changes in the industry. We do not operate in a static environment - it continuously evolves. Therefore, it is crucial for the model to accurately reflect these changes and remain current. There are two primary strategies to enhance the model's capabilities. The first is to refine the focus areas (and sub-areas), which involves addressing various security aspects more effectively. Although the width and depth of the model have been expanded through numerous iterations, there is likely still potential to enhance coverage in specific areas. Secondly, as the environment changes, it becomes

necessary to update the model to maintain its relevance over time and prevent it from becoming obsolete or an outdated self-assessment tool.

5.5.2 Tightening the alignment with standardized frameworks

The CICDSecMM currently incorporates a broad range of frameworks and standards, such as NIST, CISA, and CIS, along with contributions from industry experts. However, there is potential for future research to further refine the model, ensuring it aligns more closely with well-established frameworks that emphasize security controls, rather than continuing with its current more fragmented implementation. This could involve integrating specific security controls into the CICDSecMM to help organizations achieve compliance with certain standardized and industry-respected frameworks. By aligning the maturity model with a concrete framework, it would enhance the model's utility in supporting compliance efforts.

5.5.3 Enriching the maturity progression

Lastly, there is also an opportunity to enhance the alignment between sub-areas and practices where an absence of practices exist, specifically where security controls are lacking within a certain maturity level of a particular sub-area. While some cases have been deemed unnecessary by industry experts, addressing the absence of practices would contribute to a more comprehensive model overall.

5.5.4 Testing the effectiveness of using the model for enhancing CI/CD security capabilities

The case study evaluation conducted in this project did only evaluate the assessment, so there is currently no empirical evidence that the artifact effectively supports improvement of CI/CD security capabilities. A case study following teams or organizations over a period of time could be conducted to identify the artifact’s effectiveness in the context of enhancing CI/CD security capabilities. In such a study, the participants can use the maturity model to work with improving their capabilities based on the maturity assessment results. Based on the findings of the case study, the model would either be validated as effective for enhancing CI/CD security capabilities or it could be redesigned to better support enhancement.

5.5.5 Adding more advanced features to the maturity model

Several of the informants suggested features which were not possible to implement due to the time constraints of the project. Some of them said that it would be useful to be able to compare your organization with organizations with similar characteristics (sector, size, etc.). This would require server storage of assessment results, but the feature would provide a good indicator on how the organization is doing compared to its competitors. Another suggestion we got was full or partial automation of the assessment. Since many of the practices in the maturity model are related to the technology (platforms, tools, environments), it would be possible to write code/scripts that could retrieve some of the relevant information for the assessment. There have also been some suggestions regarding guidance on prioritization of

improvements. It would be helpful for the organizations to get guidance on which of the practices should be prioritized to enhance the security and achieve a higher level of maturity.

5.6 Limitations

The summative evaluation of the artifact was conducted through a case study with only one case (organization). With a larger sample of organizations testing the model, it would be possible to give stronger conclusions when it comes to the model's utility and relevance.

The majority of organizations that contributed to this project were Norwegian or the Norwegian branch of a multinational organization. Therefore, there is a lack of more heterogeneity among the contributing organizations to be able to claim that the artifact is generalized and applicable to any organization with CI/CD pipelines.

With a less stringent time frame, we could have allocated more time for each iteration. This could also have influenced the recruitment of informants, as busy experts do not always have time in the periods we invite them to contribute. As an example, we gave a one-week deadline on answering an email sent to the ten informants from iteration 1 and 2 in addition to another expert. Only four out of eleven informants had the time to answer and give some feedback, and only one of them managed to answer within the deadline. If we had more time than the five months we had on this project, we would probably be able to recruit more experts and get a higher response rate.

6 Conclusion

The aim of this master's thesis was to design a maturity model that encompasses CI/CD security. We initially identified a need for an artifact that effectively can measure organizations' CI/CD security. After the problem was identified and scoped, we designed, evaluated, and released the CI/CD Security Maturity Model. The purpose of the model is to provide an objective assessment tool which can serve as a means for communicating the status of CI/CD security maturity across the organizational hierarchy. We also wanted the model to be a useful starting point when implementing improvement initiatives in the maturation journey. However, the efficacy of using the maturity model for improving the CI/CD security maturity still remains to be empirically investigated.

Following well-recognized methods and requirements for design science research and maturity model design, we designed the CI/CD Security Maturity Model in a time frame of five months. By utilizing both the knowledge base and domain experts in the application environment, we were able to derive a comprehensive set of security practices and place them into appropriate maturity levels. Through a case study evaluation, we got a positive indication regarding the relevance and utility of the designed artifact. Even though there still are several initiatives that can be taken for further design and evaluation of the model, the released version of the MM constitutes a novel artifact and a good foundation for further development.

In conclusion, the thesis introduces a comprehensive CI/CD Security Maturity Model designed to serve as a benchmarking tool for organizations. Furthermore, it lays the groundwork for ongoing refinement and empirical validation, inviting future exploration within the field.

7 References

Adams, R. J., Smart, P., & Huff, A. S. (2016). Shades of Grey: Guidelines for Working with the Grey Literature in Systematic Reviews for Management and Organizational Studies. *International Journal of Management Reviews*, *19*(4), 432-454. https://doi.org/10.1111/ijmr.12102

Adekunle, S. A., Aigbavboa, C., Ejohwomu, O., Ikuabe, M., & Ogunbayo, B. (2022). A Critical Review of Maturity Model Development in the Digitisation Era. *Buildings*, *12*(6). https://doi.org/10.3390/buildings12060858

Ahmadvand, M., Pretschner, A., Ball, K., & Eyring, D. (2018). Integrity Protection Against Insiders in Microservice-Based Infrastructures: From Threats to a Security Framework. In: Mazzara, M., Ober, I., Salaün, G. (eds) *Software Technologies: Applications and Foundations*. *STAF 2018*. *Lecture Notes in Computer Science*, vol 11176. Springer. https://doi.org/10.1007/978-3-030-04771-9_43

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

Angermeir, F., Voggenreiter, M., Moyón, F., & Mendez, D. (2021). Enterprise-Driven Open Source Software: A Case Study on Security Automation. *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, Spain. https://doi.org/10.1109/ICSE-SEIP52600.2021.00037

Atlassian. (n.d.). DevOps Maturity Model. https://www.atlassian.com/solutions/devops/maturity-model

Bajpai. P., & Kannavara. R. (2023). Misplaced Trust: The Security Flaw in Modern Code Signing Process. *2023 IEEE Secure Development Conference (SecDev)*, USA. https://doi.org/10.1109/SecDev56634.2023.00018

Bajpai, P., & Lewis, A. (2022). Secure Development Workflows in CI/CD Pipelines. *2022 IEEE Secure Development Conference (SecDev)*, USA. https://doi.org/10.1109/SecDev53368.2022.00024

Becker, J., Knackstedt, R., & Pöppelbuß, J. (2009). Developing Maturity Models for IT Management. *Business & Information Systems Engineering*, *1*(3), 213-222. https://doi.org/10.1007/s12599-009-0044-5

Bekk. (n.d.). A Maturity Model for Continuous Delivery. https://bekkopen.github.io/maturity-model/

Boote, J., Erlikhman, E., Hutchison, B., Lyman, M., & Migues, S. (2023). *BSIMM14 Report 2023*. Synopsis. https://www.synopsys.com/content/dam/synopsys/sig-assets/reports/bsimm-report.pdf

Brasoveanu, R., Karabulut, Y., & Pashchenko, I. (2022). Security Maturity Self-Assessment Framework for Software Development Lifecycle. *Proceedings of the 17th International Conference on Availability, Reliability and Security*, Austria. https://doi.org/10.1145/3538969.3543806

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software, 80*(4), 571-583. https://doi.org/10.1016/j.jss.2006.07.009

Brukman, A. (2023, April 6). DevOps threat matrix. *Microsoft*. https://www.microsoft.com/en-us/security/blog/2023/04/06/devops-threat-matrix/

Center for Internet Security. (2022). CIS Software Supply Chain Security Guide. https://www.cisecurity.org/insights/white-papers/cis-software-supply-chain-security-guide

Chandramouli, R., Kautz, F., & Torres-Arias, S. (2024). *Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines* (NIST SP 800-204D). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-204D

Chau, C., Foster, M., & Sengupta, S. (2023). *Getting Started With CI/CD Pipeline Security*. DZone. https://www.redhat.com/en/resources/ci-cd-pipeline-security-dzone-analyst-material

Chickowski, E. (2023, November 14). 8 CI/CD security best practices: Protect your software pipeline. *ReversingLabs*. https://www.reversinglabs.com/blog/8-cicd-security-best-practices-software-pipeline

Cloud Native Computing Foundation. (2021). Software Supply Chain Best Practices. https://project.linuxfoundation.org/hubfs/CNCF_SSCP_v1.pdf

Codefresh. (n.d.). CI/CD Security: 7 Risks and What You Can Do About Them. https://codefresh.io/learn/ci-cd/ci-cd-security-7-risks-and-what-you-can-do-about-them/

Dancuk, M. (2021, August 26). CI/CD Security – How to Secure Your CI/CD Pipeline. *phoenixNAP*. https://phoenixnap.com/kb/ci-cd-security

de Bruin, T., Freeze, R., Kulkarni, U., & Rosemann, M. (2005). Understanding the Main Phases of Developing a Maturity Assessment Model. *Australasian Conference on Information Systems*, Australia.

Deleersnyder, S., & Win, B. D. (n.d.). *OWASP SAMM Version 2*. OWASP. https://drive.google.com/file/d/1cI3Qzfrly_X89z7StLWI5p_Jfqs0-OZv/view

Diglio, A., & Wang, J. (2023). *Secure Supply Chain Consumption Framework (S2C2F) Simplified Requirements*. Microsoft & OpenSSF. https://github.com/ossf/s2c2f/blob/main/specification/README.md

Garousi, V., Felderer, M., & Mäntylä, M. V. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, *106*, 101-121. https://doi.org/10.1016/j.infsof.2018.09.006

Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2013). Seeking Qualitative Rigor in Inductive Research: Notes on the Gioia Methodology. *Organizational Research Methods*, *16*(1), 15-31. https://doi.org/10.1177/1094428112452151

Giustini, D. (2010). Finding the hard to finds: searching for grey (Gray) literature. https://web.archive.org/web/20130608160123/http://www.slideshare.net/giustinid/finding-the-hard-to-finds-searching-for-grey-gray-literature-2010

Gregor, S. & Hevner, A. R. (2013). Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*, *37*(2), 337-355. https://www.jstor.org/stable/43825912

Gu, Y., Ying, L., Chai, H., Qiao, C., Duan, H., & Gao, X. (2023). Continuous Intrusion: Characterizing the Security of Continuous Integration Services. *2023 IEEE Symposium on Security and Privacy (SP)*, USA. https://doi.org/10.1109/SP46215.2023.10179471

Hastings, T., & Walcott, K. R. (2022). Continuous Verification of Open Source Components in a World of Weak Links. *2022 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, USA. https://doi.org/10.1109/ISSREW55968.2022.00068

Helgesson, Y. Y. L., Höst, M., & Weyns, K. (2012). A review of methods for evaluation of maturity models for process improvement. *Journal of Software: Evolution and Process*, *24*(4), 436-454. https://doi.org/10.1002/smr.560

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105. https://doi.org/10.2307/25148625

Hornbeek, M., & Jones, A. (n.d.). *A Roadmap to Continuous Delivery Pipeline Maturity*. AWS. https://pages.awscloud.com/rs/112-TZM-766/images/A-Roadmap-to-Continuous-Delivery-Pipeline-Maturity-dev-whitepaper.pdf

Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

Koishybayev, I., Nahapetyan, A., Zachariah, R., Muralee, S., Reaves, B., Kapravelos, A., & Machiry, A. (2022). Characterizing the Security of Github CI Workflows. *Proceedings of the 31st USENIX Security Symposium*, USA. https://www.usenix.org/conference/usenixsecurity22/presentation/koishybayev

Krivelevich, D., & Gil, O. (2022). *Top 10 CI/CD Security Risks*. OWASP. https://github.com/OWASP/www-project-top-10-ci-cd-security-risks/raw/3204d6d181e2a5517ffdcbe208fb536b9cc6c50b/assets/OWASP_Top_10_CICD_Risks.pdf

Kulanov, S., & Stepanov, O. (2023, November 30). Elevating CI/CD Security with Supply Chains. *SolutionsHub*. https://solutionshub.epam.com/blog/post/ci_cd_security

Kumar, R., & Goyal, R. (2021). When Security Meets Velocity: Modeling Continuous Security for Cloud Applications using DevSecOps. In: Raj, J.S., Iliyasu, A.M., Bestak, R., Baig, Z.A. (eds) *Innovative Data Communication Technologies and Application. Lecture Notes on Data Engineering and Communications Technologies*, vol 59. Springer. https://doi.org/10.1007/978-981-15-9651-3_36

Kumar, A., Nadeem, M., & Shameem, M. (2023). Prioritization of DevOps Maturity models using Fuzzy TOPSIS. *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, Finland. https://doi.org/10.1145/3593434.3594241

Lahrmann, G., Marx, F., Mettler, T., Winter, R., & Wortmann, F. (2011). Inductive Design of Maturity Models: Applying the Rasch Algorithm for Design Science Research. In: Jain, H., Sinha, A.P., Vitharana, P. (eds) *Service-Oriented Perspectives in Design Science Research. DESRIST 2011. Lecture Notes in Computer Science*, vol 6629. Springer. https://doi.org/10.1007/978-3-642-20633-7_13

Larios-Vargas, E., Elazhary, O., Yousefi, S., Lowlind, D., Vliek, M. L. W., & Storey, M.-A. (2022). DASP: A Framework for Driving the Adoption of Software Security Practices. *arXiv*. https://doi.org/10.48550/arXiv.2205.12388

Lefebvre, C., Manheimer, E., & Glanville, J. (2008). Searching for Studies. In J. P. T. Higgins, & S. Green (Eds.), *Cochrane Handbook for Systematic Reviews of Interventions: Cochrane Book Series*. Cochrane Collaboration. https://doi.org/10.1002/9780470712184.ch6

Leppänen, T., Honkaranta, A., & Costin, A. (2022). Trends for the DevOps Security. A Systematic Literature Review. In: Shishkov, B. (eds) *Business Modeling and Software Design. BMSD 2022. Lecture Notes in Business Information Processing*, vol 453. Springer. https://doi.org/10.1007/978-3-031-11510-3_12

Maayan, G. D. (n.d.). DevOps Security Challenges and How to Overcome Them. *CCSI*. https://www.ccsinet.com/blog/devops-security-challenges/

Maier, A., Moultrie, J., & Clarkson, P. J. (2009). Developing maturity grids for assessing organisational capabilities: Practitioner guidance. *4th International Conference on Management Consulting: Academy of Management*, Austria.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251-266. https://doi.org/10.1016/0167-9236(94)00041-2

Martin, R. A. (2020). Visibility & Control: Addressing Supply Chain Challenges to Trustworthy Software-Enabled Things. *2020 IEEE Systems Security Symposium (SSS)*, USA. https://doi.org/10.1109/SSS47320.2020.9174365

Martínez, J., & Durán, J. M. (2021). Software Supply Chain Attacks, a Threat to Global Cybersecurity: SolarWinds' Case Study. *International Journal of Safety and Security Engineering*, *11*(5), 537-545. https://doi.org/10.18280/ijsse.110505

Microsoft. (n.d.). What are the Microsoft SDL practices? https://web.archive.org/web/20240114155511/https://www.microsoft.com/en-us/securityengineering/sdl/practices

Moghnie, S., Niedzialkowski, T., & Sehgal, S. (2020). *The Six Pillars of DevSecOps: Automation*. Cloud Security Alliance & SAFECode. https://cloudsecurityalliance.org/artifacts/devsecops-automation

Morgenstern, T. (2023, December 10). CI/CD security – 5 best practices. *Vulcan Cyber*. https://vulcan.io/blog/ci-cd-security-5-best-practices/

Moriconi, F., Neergaard, A. I., Georget, L., Aubertin, S., & Francillon, A. (2023). Reflections on Trusting Docker: Invisible Malware in Continuous Integration Systems. *2023 IEEE Security and Privacy Workshops (SPW)*, USA. https://doi.org/10.1109/SPW59333.2023.00025

Myers, M. D. & Avison, D. (2002). *Qualitative Research in Information Systems*. SAGE Publications Ltd. https://uk.sagepub.com/en-gb/eur/qualitative-research-in-information-systems/book205159

Nalini, M. K., Mahalakshmi, B. S., Khandelwal, N., Pai, N., & Sharan, L. (2023). CI/CD Pipeline with Vulnerability Mitigation. *2023 International Conference on Recent Advances in Science and Engineering Technology (ICRASET)*, India. https://doi.org/10.1109/ICRASET59632.2023.10419921

National Security Agency & Cybersecurity and Infrastructure Security Agency. (2023). *Defending Continuous Integration/Continuous Delivery (CI/CD) Environments*. https://media.defense.gov/2023/Jun/28/2003249466/-1/-1/0/CSI_DEFENDING_CI_CD_ENVIRONMENTS.PDF

National Security Agency, Office of the Director of National Intelligence, & Cybersecurity and Infrastructure Security Agency. (2023). *Securing the Software Supply Chain: Recommended Practices for Managing Open-Source Software and Software Bill of Materials*. Enduring Security Framework. https://media.defense.gov/2023/Dec/11/2003355557/-1/-1/0/ESF_SECURING_THE_SOFTWARE_SUPPLY_CHAIN%20RECOMMENDE

D%20PRACTICES%20FOR%20MANAGING%20OPEN%20SOURCE%20SOFTWARE%20AND%20SOFTWARE%20BILL%20OF%20MATERIALS.PDF

Neharika, K., & Lennon, R. G. (2023). Investigations into Secure IaC Practices. In: Yang, XS., Sherratt, S., Dey, N., Joshi, A. (eds) *Proceedings of Seventh International Congress on Information and Communication Technology. Lecture Notes in Networks and Systems*, vol 448. Springer. https://doi.org/10.1007/978-981-19-1610-6_25

Ng, J. [@julie-ng], Buck, A. [@alexbuckgit], Zimmergren, T. [@Zimmergren], Kim, C. [@RedZephyr13], & @v-chmccl. (2022, July 8). Securing the pipeline and CI/CD workflow. *Microsoft Learn*. https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/secure/best-practices/secure-devops

NIS1 Directive. (2016). *Directive (EU) 2016/1148 of the European Parliament and of the Council of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union*. EUR-Lex. https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32016L1148

Open Worldwide Application Security Project. (n.d.). CI/CD Security Cheat Sheet. https://cheatsheetseries.owasp.org/cheatsheets/CI_CD_Security_Cheat_Sheet.html

Pagel, T. & Prasad, A. (n.d.). OWASP Devsecops Maturity Model. *OWASP*. https://owasp.org/www-project-devsecops-maturity-model/

Palo Alto Networks. (n.d.). What Is the CI/CD Pipeline? https://www.paloaltonetworks.com/cyberpedia/what-is-the-ci-cd-pipeline-and-ci-cd-security

Pan, Z., Shen, W., Wang, X., Yang, Y., Chang, R., Liu, Y., Liu, C., Liu, Y., & Ren, K. (2024). Ambush From All Sides: Understanding Security Threats in Open-Source Software CI/CD Pipelines. *IEEE Transactions on Software Engineering*, *21*(1), 403-418. https://doi.org/10.1109/TDSC.2023.3253572

Patra, M. K., Kumari, A., Sahoo, B., & Turuk, A. K. (2022). Docker Security: Threat Model and Best Practices to Secure a Docker Container. *2022 IEEE 2nd International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC)*, India. https://doi.org/10.1109/iSSSC56467.2022.10051481

Pecka, N., Othmane, L. B., & Valani, A. (2022). Privilege Escalation Attack Scenarios on the DevOps Pipeline Within a Kubernetes Environment. *Proceedings of the International Conference on Software and System Processes and International Conference on Global Software Engineering*, USA. https://doi.org/10.1145/3529320.3529325

Rafi, S., Yu, W., Akbar, M. A., Alsanad, A., & Gumaei, A. (2020). Prioritization Based Taxonomy of DevOps Security Challenges Using PROMETHEE. *IEEE Access*, *8*, 105426-105446. https://doi.org/10.1109/ACCESS.2020.2998819

Rahman, A., Parnin, C., & Williams, L. (2019). The Seven Sins: Security Smells in Infrastructure as Code Scripts. *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, Canada. https://doi.org/10.1109/ICSE.2019.00033

Rahman, A., Barsha, F. L., & Morrison, P. (2021a). Shhh!: 12 Practices for Secret Management in Infrastructure as Code. *2021 IEEE Secure Development Conference (SecDev)*, USA. https://doi.org/10.1109/SecDev51306.2021.00024

Rahman, A., Rahman, M. R., Parnin, C., & Williams, L. (2021b). Security Smells in Ansible and Chef Scripts: A Replication Study. *ACM Transactions on Software Engineering and Methodology*, *30*(1). https://doi.org/10.1145/3408897

Rajapakse, R. N., Zahedi, M., & Babar, M. A. (2021). An Empirical Analysis of Practitioners' Perspectives on Security Tool Integration into DevOps. *Proceedings of the 15th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*, Italy. https://doi.org/10.1145/3475716.3475776

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

Rangnau, T., Buijtenen, R., Fransen, F, & Turkmen, F. (2020). Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines. *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*, Netherlands. https://doi.org/10.1109/EDOC49727.2020.00026

Rehn, A., Boström, P., & Palmborg, T. (2013, February 06). The Continuous Delivery Maturity Model. *InfoQ*. https://www.infoq.com/articles/Continuous-Delivery-Maturity-Model/

Schöpfel, J. & Farace, D. J. (2009). Grey Literature. In M. J. Bates, & M. N. Maack (Eds.), *Encyclopedia of Library and Information Sciences* (3rd ed., pp. 2029-2039). CRC Press.

Scovetta, M. (2020). Threats, Risks, and Mitigations in the Open Source Ecosystem. https://github.com/ossf/wg-metrics-and-metadata/blob/main/publications/threats-risks-mitigations/v1.2/Threats%2C%20Risks%2C%20and%20Mitigations%20in%20the%20Open%20Source%20Ecosystem.md

Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. *IEEE Access*, *5*, 3909-3943. https://doi.org/10.1109/ACCESS.2017.2685629

Shamim, M. S. I., Bhuiyan, F. A., & Rahman, A. (2020). XI Commandments of Kubernetes Security: A Systematization of Knowledge Related to Kubernetes Security Practices. 2020 IEEE Secure Development (SecDev), USA. https://doi.org/10.1109/SecDev45635.2020.00025

Shevchuk, R., Karpinski, M., Kasianchuk, M., Yakymenko, I., Melnyk, A., & Tykhyi, R. (2023). Software for Improve the Security of Kubernetes-based CI/CD Pipeline. *2023 13th International Conference on Advanced Computer Information Technologies (ACIT)*, Poland. https://doi.org/10.1109/ACIT58437.2023.10275654

Souppaya, M., Morello, J., & Scarfone, K. (2017). *Application Container Security Guide* (NIST SP 800-190). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-190

Souppaya, M., Scarfone, K., & Dodson, D. (2022). *Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities* (NIST SP 800-218). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-218

Springett, S. (2020). *Software Component Verification Standard Version 1.0*. OWASP. https://github.com/OWASP/Software-Component-Verification-Standard/releases/download/1.0/OWASP_SCVS-1.0-en.pdf

Supply-chain Levels for Software Artifacts. (n.d.). Get started. https://slsa.dev/get-started

Sysdig. (n.d.). CI/CD Security: Securing Your CI/CD Pipeline. https://sysdig.com/learn-cloud-native/container-security/cicd-pipeline/

Tak, B., Isci, C., Duri, S., Bila, N., Nadgowda, S., & Doran, S. (2017). Understanding Security Implications of Using Containers in the Cloud. *Proceedings of the 2017 USENIX Annual Technical Conference (USENIX ATC '17)*, USA. https://www.usenix.org/conference/atc17/technical-sessions/presentation/tak

The Hacker News. (2023). CI/CD Risks: Protecting Your Software Development Pipelines. https://thehackernews.com/2023/11/cicd-risks-protecting-your-software.html

Thuan, N. H., Drechsler, A., & Antunes, P. (2019). Construction of Design Science Research Questions. *Communications of the Association for Information Systems*, *44*, 332-363. https://doi.org/10.17705/1CAIS.04420

Tokerud, S., & Jansen, J. N. (2022). *Designing the Extended Zero Trust Maturity Model: A Holistic Approach to Assessing and Improving an Organization's Maturity Within the Technology, Processes and People Domains of Information Security* [Master's thesis, University of Agder]. AURA research archive. https://hdl.handle.net/11250/3019801

Tokerud, S., Jansen, J. N., Niemimaa, M., & Järveläinen, J. (2023). Designing Extended Zero Trust Maturity Model – From Technical to Socio-Technical. *Rising like a Phoenix: Emerging from the Pandemic and Reshaping Human Endeavors with Digital Technologies ICIS 2023*, India. https://aisel.aisnet.org/icis2023/cyber_security/cyber_security/5

Vakhula, O., Opirskyy, I., & Mykhaylova, O. (2023). Research on Security Challenges in Cloud Environments and Solutions based on the "Security-as-Code" Approach.

CPITS-2023-II: Cybersecurity Providing in Information and Telecommunication Systems, Ukraine. https://ceur-ws.org/Vol-3550/paper5.pdf

van Steenbergen, M., Bos, R., Brinkkemper, S., van de Weerd, I., & Bekkers, W. (2010). The Design of Focus Area Maturity Models. In: Winter, R., Zhao, J.L., Aier, S. (eds) *Global Perspectives on Design Science Research*. Springer. https://doi.org/10.1007/978-3-642-13335-0_22

Vasile. T., Cane. S., Bertram. C. & Jakob. F. (2019). Applying Security Concepts to Continuous Integration for the Purpose of Testing Embedded Systems. *AmE 2019 - Automotive meets Electronics; 10th GMM-Symposium*, Germany. https://ieeexplore.ieee.org/document/8727844

Venable, J., Pries-Heje, J. & Baskerville, R. (2016). FEDS: a Framework for Evaluation in Design Science Research. *European Journal of Information Systems*, *25*(1), 77-89. https://doi.org/10.1057/ejis.2014.36

Veritis. (n.d.). DevOps Maturity Model – From Traditional IT to Complete DevOps. https://www.veritis.com/blog/meet-full-devops-potential-with-devops-maturity-model/

Vlietland, J. (2019, June 14). Continuous Delivery 3.0 Maturity Model (CD3M). *National Institute for the Software Industry*. https://nisi.nl/continuousdelivery/articles/maturity-model

Wanden-Berghe, C. & Sanz-Valero, J., (2012). Systematic reviews in nutrition: standardized methodology. *British Journal of Nutrition*, *107*(S2), S3-S7. https://doi.org/10.1017/S0007114512001432

Xiao, Y., & Watson, M. (2019). Guidance on Conducting a Systematic Literature Review. *Journal of Planning Education and Research*, *39*(1), 93-112. https://doi.org/10.1177/0739456x17723971

Yazdani, A., & Thakur, M. S. (n.d.). OWASP DevSecOps Guideline. *OWASP*. https://owasp.org/www-project-devsecops-guideline/

Zampetti, F., Geremia, S., Bavota, G., & Penta, M. D. (2021). CI/CD Pipeline Evolution and Restructuring: A Qualitative and Quantitative Study. *2021 IEEE International Conference on Software Maintenance and Evolution(ICSME)*, Luxembourg. https://doi.org/10.1109/ICSME52107.2021.00048

Zeini, A., Lennon, R. G., & Lennon, P. (2023). Preliminary Investigation into a Security Approach for Infrastructure as Code. In: Yang, XS., Sherratt, R.S., Dey, N., Joshi, A. (eds) *Proceedings of Eighth International Congress on Information and Communication Technology. ICICT 2023. Lecture Notes in Networks and Systems*, vol 694. Springer. https://doi.org/10.1007/978-981-99-3091-3_63

Zhou, X., Mao, R., Zhang, H., Dai, Q., Huang, H., Shen, H., Li, J., & Rong, G. (2023). Revisit security in the era of DevOps: An evidence-based inquiry into DevSecOps industry. *IET Software*, *17*(4), 435-454. https://doi.org/10.1049/sfw2.12132

Appendix A: Interview guide

CI/CD Security Maturity Model - Intervjuguide

Vi introduserer oss selv og forklarer prosjektet og modellen.

Vi spør om deltakeren samtykker til opptak av møtet.

Del 1: Generell info om intervjuobjektet (bakgrunn og erfaringer).

1. Hvor gammel er du?

2. Har du en utdannelse og eventuelt hvilken?

3. Hvilken stilling har du i dag?

4. Hvor lenge har du jobbet med CI/CD?

5. Hvilken rolle har du hatt i de sammenhengene du har jobbet med CI/CD?
    a. Har du vært direkte involvert i sikkerhetsarbeid med CI/CD?

6. Har du kjennskap til modenhetsmodeller?
    a. Hvis ja:
        i. Har du benyttet noen tidligere?
        ii. Kjenner du til noen sikkerhetsrettede modenhetsmodeller?
            1. Kan du navngi de du kjenner?
            2. Har du brukt noen (sikkerhetsrettede)?
        iii. Hvilket inntrykk har du av modenhetsmodeller? (hva er din erfaring)

7. Har du noen eksempler på situasjoner hvor du har opplevd dårlig CI/CD-sikkerhet?
    a. Klarer du å forklare hvorfor det ble sånn?
    b. Hva var betingelsene/forholdene, og hvordan påvirket dette sikkerheten?

8. Har du noen eksempler på situasjoner hvor du har opplevd god CI/CD-sikkerhet?
    a. Klarer du å forklare hvorfor det ble sånn?
    b. Hva var betingelsene/forholdene, og hvordan påvirket dette sikkerheten?

9. Har du noen gang opplevd utfordringer med å implementere sikkerhet i CI/CD?
    a. Hva var utfordrende?
    b. Hvorfor var det utfordrende?
    c. Har du lært noe fra dette?

Modellspørsmål:

1. Har du valgt deg ut noen fokusområder som du har fokusert på?
    1.1. Hvilke?
    1.2. Hvorfor valgte du disse?
    Spesifikt gå gjennom hvert område og spør:
    1.3. Hva synes du om underområdene?
    1.4. Hva synes du om praksisene for hvert underområde?
        1.4.1. Går gjennom level 1, 2 og 3 (Praksis + Plassering)
        1.4.2. Diskuterer potensielle praksiser for hvert nivå (level)

2. Hva synes du om abstraksjonsnivået på praksiser? (Konkret vs. Abstrakt)
    2.1. Er det deler av modellen som du mener kan ha nytte av et høyere eller lavere abstraksjonsnivå? (hvis ja: oppfølg med hvilke og hvorfor)
    2.2. Er det noen praksiser som blir for spesifikke/konkrete? (hvis ja: oppfølg med hvilke og hvorfor)
    2.3. Er det noen praksiser som er for kontekst-avhengige og ikke er universelle nok? (hvis ja: oppfølg med hvilke og hvorfor)

3. Synes du alle fokusområdene er relevante for sikkerhet i CI/CD?
    3.1. Hva synes du om inndelingen av fokusområder?
        3.1.1. Er det noe som overlapper?
        3.1.2. Er det noe du ville omrokert/omstrukturert?
    3.2. Er det noen viktige sikkerhetsaspekter du mener mangler i denne modellen?

4. Ser du noen områder i modellen som kunne vært definert klarere eller mer detaljert?

5. Hva synes du om praksisenes plassering i modenhetsnivåer? (Generelt)
    5.1. Er det noen praksiser som er plassert for høyt eller lavt?
    5.2. Klarer du å foreslå noen praksiser som kan fylles inn i de tomme cellene?

6. Hvordan vil du vekte de ulike Fokus-områdene (og sub-områdene)
    6.1.Eks: 20% av områdene står for 80% av "sikkerheten". (litt urealistisk eksempel, men ment for å illustrere)

7. Hvilke kvaliteter anser du som viktige for den ferdigstilte modellen?
    7.1.Hva skal til for at den skal være nyttig og gi verdi og støtte?

8. Åpent spørsmål: Har du noen andre tilbakemeldinger du vil dele med oss?

Spørsmål mot slutten:

Kunne du tenke deg å delta i en ny runde når vi har implementert forbedringer av modellen?

Kjenner du noen du mener vi burde snakke med / involvere?

Appendix B: Information letter

Would you like to participate in the research project:

Designing a Maturity Model for Assessing and Enhancing Security in CI/CD Pipelines

Purpose with this project
This is a question for you about whether you would like to participate in a master's project where the purpose is to explore focus areas and practices within security in Continuous Integration and Continuous Delivery/Deployment (CI/CD). We aim to design a maturity model that will help organizations evaluate and improve their security posture/condition and maturity. To achieve this, we need information from people who have experience within the problem area and are willing to share their insights and knowledge.

Furthermore, we want to investigate the research question:
How can a maturity model be designed to facilitate evaluation and enhancement of the security posture of CI/CD pipelines?

The only purpose for which collected personal information will be used is our master's project.

Why are you asked to participate:
You are receiving this request because

We are reaching out to participants whom we believe can provide valuable insights from various perspectives, with the goal of establishing a solid data foundation for further analysis of our findings. We plan to interview between 10 and 15 people. The selection of interview subjects can occur in several ways. This could be, for example, through searches on LinkedIn, activity in different forums, or recommendations from other individuals.

Who is responsible for the research project?
We are two master's students (Nicolai Knudsen Bjørntvedt and Knut Sæther) from the University of Agder at the Faculty of Social Sciences - Department of Information Systems who will be responsible for the project. The University of Agder is thus the institution responsible for the personal data processed in the project.

Participation is voluntary
Participation in the project is voluntary. There will be no negative consequences for you if you choose not to participate or decide to withdraw later.

What does participation entail for you?

Our method of data collection is interviews, with digital recording of audio and video. The scope will cover various areas and practices within security in Continuous Integration and Continuous Delivery/Deployment (CI/CD).

Personal information to be collected:

- Name
- Contact information
- Individuals in photographs or video recordings
- Voices in audio recordings
- Background information, which in combination could identify an individual

Video and audio recordings will be used during the interview. This provides us with the opportunity to focus on the conversation and the interview itself, rather than getting bogged down in taking extensive notes.

Brief overview of privacy:
We will only use the information about you for the purposes we have outlined in this document. We handle personal information confidentially and in accordance with privacy regulations. You can read more about privacy on the next page.

Detailed information about privacy - how we store and use your information:
Only the students Nicolai Knudsen Bjørntvedt and Knut Sæther in the Master's Program in Cyber Security - Security Management will have access to the raw material from the interview(s). All data collected through the interviews will be anonymized, and personal information will not be transcribed or included in the documentation. Participants will not be identifiable in the publication.

The data will be stored on UiA's tenant on OneDrive (cloud service).

What gives us the right to process personal information about you?
We process information about you based on your consent.

On behalf of Nicolai Knudsen Bjørntvedt and Knut Sæther, the privacy services at Sikt - Norwegian Agency for Shared Services in Education and Research, have assessed that the processing of personal data in this project is in accordance with privacy regulations.

Your rights:
As long as you can be identified in the data material, you have the right to:

- Request access to the information we process about you and receive a copy of the information,
- Have incorrect or misleading information about you corrected,
- Have personal information about you deleted,
- Lodge a complaint with the Data Protection Authority about the processing of your personal information.

We will provide you with a justification if we believe that you cannot be identified or that the rights cannot be exercised.

What happens to your personal information when the research project is concluded?

The project is scheduled to conclude on June 7, 2024.

After the delivery of the master's thesis, all of our material containing your personal information will be deleted on the same day.

Questions:
If you have any questions or wish to exercise your rights, please contact:

Supervisors:

Name: Paolo Spagnoletti

Position: Associate Professor

E-mail: paolo.spagnoletti@uia.no

Phone: +47 38142624

Name: Marko Ilmari Niemimaa

Position: Associate Professor

E-mail: marko.niemimaa@uia.no

Phone: +47 38141842

Students:

Name: Nicolai Knudsen Bjørntvedt

E-mail: nicolaikb@uia.no

Name: Knut Sæther

E-mail: knut.sather@uia.no

Data Protection Officer (UiA): Trond Hauso personvernombud@uia.no (+47 936 01 625)

If you have questions regarding Sikt's assessment of the project, you can send an e-mail to: personverntjenester@sikt.no, or call: +47 73 98 40 40.

Best regards,

Project supervisor	Students
Paolo Spagnoletti (Associate Professor)	Nicolai Knudsen Bjørntvedt and Knut Sæther

---

Declaration of consent
I have received and understood the information about the project *Designing a Maturity Model for Assessing and Enhancing Security in CI/CD Pipelines*, and I have had the opportunity to ask questions. I consent to:

- ☐ participate in **the interview**

I consent to my information being processed until the project is completed.

(Signed by project participant, date)

Appendix C: A comparison of maturity model design methodologies

Maturity models have been subject for critique when it comes to the documentation of the design process. The critique can be divided into several aspects, such as a lack of quality documentation of the design process (or any documentation at all) and designing a MM as the researchers chose – without following a verifiable approach (Adekunle et al., 2022; Becker et al., 2009). Thus, we synthesized a maturity model design process consisting of 4 common phases identified in the past literature describing maturity model design methodologies.

Common phases	**Becker et al. (2009)**	**de Bruin et al. (2005)**	**Lahrmann et al. (2011)**	**van Steenbergen et al. (2010)**
Preparative phase	1. Problem definition 2. Comparison of existing maturity models 3. Determination of development strategy	1. Scope	1. Identify need or new oppurtunity 2. Define scope	1. Scope
Design phase	4. Iterative maturity model development	2. Design 3. Populate	3. Design model	2. Design model 3. Develop instrument
Evaluation phase	5. Conception of transfer and evaluation	4. Test	4. Evaluate design	
Deployment and reporting phase	6. Implementation of transfer media 7. Evaluation 8. Rejection of maturity model	5. Deploy 6. Maintain	5. Reflect evolution	4. Implement and Exploit

Table: Comparative overview of common design phases

Preparative phase

All of the four maturity model design procedure models start with one or more phases where the problem in focus or scope of the project and model is defined. Becker et al.'s (2009) three first phases can be categorized as preparative work ahead of the design phase. The three other design procedure models do as well have one or more phase(s) before the design of the maturity model. What is common for all four procedure models is that they all start with identification and scoping of the problem domain. However, there are some differences when comparing what the different models have incorporated into the preparative phase(s). Becker et al. (2009) and van Steenbergen et al. (2010) are the only ones that mention a comparison of existing maturity models in the same or similar domains. de Bruin et al. (2005) have included identification of stakeholders to the first phase, which is unique compared to the other

models. The purpose of this specific process is to identify stakeholders from academia, industry, non-profits and government to assist in the maturity model development. Only the model by Becker et al. (2009) has incorporated determination of design strategy to the preparative phase. The determination includes the assessment of design strategies, like a completely new model design, enhancement of an existing model, combining several models into one new model, or transfer of structures or contents from existing models to new domains.

Design phase

The second of the common phases is the design phase. This phase involves the development and design of the maturity model. Becker et al. (2009) is the most specific and strict procedure model when it comes to how to design the model, as it specifically requires iterative development. None of the other design procedure models have specified that the development is to be done in a sequential, iterative, or any other manner. However, one could argue that van Steenbergen et al. (2010) also include iterative development, as their model is composed in a way that the maturity model is supposed to be deployed before it is evaluated and iteratively improved. As such, the "Implement & exploit" phase in the procedure model of van Steenbergen et al. (2010) is not only about deploying the maturity model, but also to refine and improve it based on evaluations from the first applications of the MM. In a way, the design phase has been prolonged into the end-phase they call "Implement & exploit".

Even though Becker et al. (2009) are very specific with emphasizing iterative design, their design phase is less described in detail than the design phases in the models of de Bruin et al. (2005) and van Steenbergen et al. (2010). All of these three distinct models emphasize the usage of literature reviews and exploratory research methods for identifying the dimensions to include in the maturity model. de Bruin et al. (2005) use the term *domain components*, while van Steenbergen et al. (2010) use the term *focus areas* for what Becker et al. (2009) call dimensions. For complex domains, de Bruin et al. (2005) recommend identifying sub-dimensions. The overall goal is to arrive at dimensions and sub-dimensions that are mutually exclusive and collectively exhaustive (de Bruin et al., 2005). Additionally, van Steenbergen et al. (2010, p. 327) highlight that "[g]rouping the focus areas into a small number of categories may add to the accessibility of the model and is also a means of achieving completeness." The other design procedure models do not specifically suggest grouping the dimensions into categories.

The appropriate number of domains and sub-domains is defined differently by de Bruin et al. (2005) and van Steenbergen et al. (2010). While de Bruin et al. (2005) claim that these numbers should be kept low, van Steenbergen et al. (2010) refer to Maier et al. (2009), which claim that a number of around 20 dimensions is a good number on average. de Bruin et al. (2005) argue that keeping the number low will minimize the perceived complexity of the model and secures the independence of the dimensions.

Another important aspect during the design phase of developing a maturity model is to define maturity levels. These levels represent an evolutionary path of capabilities as progressive

maturity stages, and it is crucial that the levels are distinct and well-defined with a logical progression through the levels (de Bruin et al., 2005; van Steenbergen et al., 2010). The process of defining maturity levels can follow either a top-down or bottom-up approach (de Bruin et al., 2005). de Bruin et al. (2005) suggest that level definitions should be developed, as a means of expanding level names and providing a summary of the significant requirements and measures of the level. van Steenbergen et al. (2010) specify explicitly that the capabilities within the maturity levels are to be based on literature review complemented with expert discussions.

van Steenbergen et al. (2010) go more into detail regarding the capabilities within the maturity levels. They emphasize that there must be an identification of dependencies between capabilities, both within the same focus area and across the different focus areas. If there are capabilities that must be in place before another capability can be implemented, this must be reflected in their placement into maturity levels. Another useful feature which can be implemented to maturity models is improvement actions (van Steenbergen et al., 2010). The purpose of improvement actions is to support practitioners in moving to the capabilities. Instead of presenting the improvement actions as prescriptions, they should be presented as suggestions.

Since a maturity model is an artifact which can be used to measure and indicate an organization's maturity level, the design phase of the development of a maturity model also includes identifying what to measure and how to measure it. Both de Bruin et al. (2005) and van Steenbergen et al. (2010) suggest formulating control questions for measurement. These questions will be based on the dimensions and the chosen capabilities within those dimensions. In addition, de Bruin et al. (2005) highlight that it is important that questions measure what they are intended to measure, and that the number of questions is balanced to ensure reliability of the data. Questionnaires are recommended for the assessment of maturity (de Bruin et al., 2005; van Steenbergen et al., 2010).

In contrast to the three other design procedure models, Lahrmann et al. (2011) suggest a quantitative approach to constructing maturity models, using the Rasch algorithm.

Evaluation phase

The population of the model marks the end of the design phase, and the start of the evaluation phase. In this phase, the model is tested for relevance and rigor (de Bruin et al., 2005). Not only the construct of the model is to be tested, but also the model instruments. What is tested is the validity, reliability and generalizability of the construct of the model and the model instruments. For construct validity, de Bruin et al. (2005) distinguish between face and content validity. Respectively, face and content validity regards the translations of the constructs compared to the identified scope of the model and the completeness of the representation of the domain. For the model instruments, the testing must ensure that the instruments measure what they were intended to measure (validity) and ensure that the obtained results are accurate and repeatable (reliability) (de Bruin et al., 2005). When the design process is following an iterative approach, the evaluation is integrated to the design

phase rather than being a standalone phase. Becker et al. (2009) describes that the result of an iteration must be tested for comprehensiveness, consistency, and problem adequacy. The sequel of the design procedure will then be decided based on the result of this evaluation. As Lahrmann et al. (2011) sensibly describes, evaluation is a crucial step in design science research projects, and the acceptance of a maturity model depends critically on its utility, validity, reliability, and generalizability.

Deployment and reporting phase

Most of the design procedure models mention evaluation as a part of the deployment phase as well. Whereas the above evaluation phase can be regarded as a pre-deployment evaluation, the evaluation described in relation to deployment can be regarded as post-deployment evaluation. For this post-deployment evaluation, the defined goals are compared with real-life observations, for instance in a case study or making the model accessible on the internet for free access (Becker et al., 2009). de Bruin et al. (2005) suggest testing and verifying the generalizability in two steps, first by using the design collaborators as respondents and then applying the model within entities independent from the model development. van Steenbergen et al. (2010) do not describe any evaluation ahead of deployment but suggest implementing the model and using the first applications of the model for evaluation. Based on this post-deployment evaluation, the model is iteratively improved.

After the model is made available for use, there is a need for maintenance and regular evaluation if it is supposed to be permanently valid (Becker et al., 2009). When the domain knowledge broadens and deepens, as a result of changing conditions, new scientific insights or technological progress, further development is needed for the model to remain valid (Becker et al., 2009; de Bruin et al., 2005; Lahrmann et al., 2011). Already in an early stage of the design process, it is important to reflect on how alterations in model design and deployment should be handled (Lahrmann et al., 2011).

Only Becker et al. (2009) and van Steenbergen et al. (2010) explicitly say that the results of the design should be communicated to the scientific community and practitioners. This is a vital element of design science research, which is why Becker et al. (2009) based on the guidelines defined by Hevner et al. (2004) included the two last requirements (R7 and R8) in their maturity model development requirements.

If the maturity model at some point after the deployment gets negative results through the regular evaluations, and redesign is considered out of the question, the model should be rejected (Becker et al., 2009). When it is considered obsolete, the best solution will be to actively take it off the market, purposefully and explicitly.

Appendix D: Focus areas – GL Article assessment

Qs..Num	Grey Literature												Notes
	GL1	GL2	GL3	GL4	GL5	GL6	GL7	GL8	GL9	GL10	GL11	GL12	
1	1	0,5	1	0	0,5	0	1	0	1	1	1	1	GL2 is a cybersecurity firm but not that reputable compared to the others. GL4 is a phone infra-provider. Gl6 and GL8 is a more general technology provider.
2	0	1	0	1	0	1	1	1	1	0	1	0	7 of the totalt 12 papers have an author.
3	1	0,5	1	0,5	0,5	1	1	0	1	1	1	1	GL2, 4 and 5 has published some relevant articles, but not completely. GL8 has not done that.
4	0,5	0,5	1	0,5	1	0,5	1	0,5	1	1	1	1	GL1, 2, 4, 6, and 8 are technology-oriented companies that explicitly advocate for their use of CI/CD pipelines, implying a certain level of expertise can be assumed.
5	1	1	1	1	1	1	1	1	1	1	1	1	All of the 12 papers have a clearly stated aim
6	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	We don't see the assessment of methodology as applicable in this context.
7	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Grey literature often uses different types of references compared to white published papers.
8	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	We don't see the assessment of limits as applicable in this context.
9	1	1	1	1	1	1	1	1	1	1	1	1	Hence it do indeed cover specific topics, not questions
10	1	1	1	1	1	1	1	1	1	1	1	1	
11	1	1	1	0,5	1	1	1	1	1	1	1	1	GL4 seems to take advantage of presenting some "selling points" towards their infrastructure services.
12	1	1	1	0,5	1	1	1	1	1	1	1	1	All of the GL seems objective, except GL4 which seems to promote its own technology.
13	1	0,5	1	0,5	0,5	1	1	1	1	1	1	1	Yes, most of the grey literature appears to be unbiased. However, GL2, 4, and 5 seem to exhibit some bias towards their own products.
14	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	We don't see this area as applicable in this context.
15	1	1	1	1	1	1	1	1	1	1	1	1	Yes, dates are on place.
16	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	We don't see this area as applicable in this context.
17	1	1	1	1	1	1	1	1	1	1	1	1	Yes, they all add something to the research
18	1	1	1	1	1	1	1	1	1	1	1	1	Strengthen the posotion.
19	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	We don't see this area as applicable in this context.
20	0,5	0,5	0,5	0,5	0,5	0,5	1	0,5	0,5	0,5	1	1	
Sum	12	11,5	12,5	11	11	12	14	11	13,5	12,5	14	13	
Normalized (0-1)	0,85	0,82	0,89	0,78	0,78	0,85	1	0,78	0,96	0,89	1	0,92	

Appendix E: CI/CD Security Practices – GL Article Assessment

Qs..Num	Grey Literature																			
	GL1	GL2	GL3	GL4	GL5	GL6	GL7	GL8	GL9	GL10	GL11	GL12	GL13	GL14	GL15	GL16	GL17	GL18	GL19	GL20
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0,5	1
2	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	1	0	1
3	0,5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0,5	1
4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
7	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
8	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
9	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1	0,5	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
14	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
15	1	1	0,5	1	1	0,5	1	1	1	1	0	0	0	0,5	0	0	0,5	1	0	1
16	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
17	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
20	0,5	0,5	0,5	1	1	1	1	1	1	0,5	1	1	1	1	1	1	1	1	0,5	1
Sum	12	12,5	12	13	13	12,5	14	14	14	13	13	13	13	13,5	13	13	12,5	14	10,5	14
Normalized (0-1)	0,85	0,89	0,85	0,92	0,92	0,89	1	1	1	0,92	0,92	0,92	0,92	0,96	0,92	0,92	0,89	1	0,75	1

Question number	Rationale
1	All of the guidelines are reputable, except for SLSA, despite its collaborations with major entities like Verizon and Google.
2	12 out of the 20 articles have an author
3	Yes, most of the authors/organizations have published other work in the field
4	All Grey Literature (GL) sources possess expertise in their respective subject areas.
5	All of the GL's have a clearly stated aim
6	Not deemed applicable for our context
7	Not deemed applicable for our context
8	Not deemed applicable for our context
9	Yes, they all cover a specific question
10	Yes, all of the GL's refers to a specific case and/or topic.
11	Yes, all of the GL's appear to be balanced in their presentation, as each is maintained by well-respected organizations and institutions that are unbiased towards specific products or services.
12	The Grey Literature (GL) generally appears to be objective in its content. However, GL 10 emphasizes that its views are based on the author's personal opinion, which is subjective. Nonetheless, it is assumed that the OpenSSF has overseen and vetted the content for accuracy.
13	No, there seems to be little vested interest in the Grey Literature (GL), as it is predominantly published by government institutions and other highly trusted sources that provide reliable information and guidelines.
14	Not deemed applicable for our context.
15	Approximately half of the grey literature clearly displays a publication date. However, it is challenging to locate and access the dates for GL 3, 6, 14, and 17. Additionally, GL 11, 12, 13, 15, 16, and 19 appear to lack a date stamp altogether.
16	Not deemed applicable for our context
17	Yes, they all add something unique to the research.
18	Yes, they all strengthen the current position
19	Not deemed applicable for our context
20	

Appendix F : CI/CD Existing Security Maturity Models – GL Article Assessment

Qs..Num	Grey Literature											Notes
	GL1	GL2	GL3	GL4	GL5	GL6	GL7	GL8	GL9	GL10	GL11	
1	0,5	0	1	1	0,5	0	0	0,5	1	1	0,5	Around half of the Grey literatures(GL's) is published by a reputable organization.
2	1	1	0	0	0	0	1	0	1	1	0,5	Around half of the Grey literature has an author
3	0,5	0	1	1	0,5	0	0,5	0,5	1	1	1	Most of the publishing organizations/authors have published other works in the field.
4	0,5	0,5	1	1	0,5	0,5	0,5	0,5	1	1	1	Most organizational and publishing institutions, as well as global organizations, are considered to have expertise in their respective fields.
5	1	1	1	1	1	1	1	1	1	1	1	Yes, all of the GL's have a clearly stated aim
6	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
7	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
8	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
9	1	1	1	1	1	1	1	1	1	1	1	Yes, they all cover a specific question/topic
10	1	1	1	1	1	1	1	1	1	1	1	Yes, they all cover a specific case/topic
11	1	1	1	1	1	0,5	1	1	1	1	1	Yes, most of the guidelines appear to be balanced in their presentation. However, guideline GL6 only presents a model without providing any additional context.
12	1	1	1	1	1	1	1	1	1	1	1	Yes, objective
13	1	1	0,5	1	1	1	1	1	1	1	1	None of the grey literature appears to have a vested interest, except for the AWS article, which tends to emphasize Amazon-based technologies as solutions to problems.
14	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
15	1	1	0,5	0,5	0,5	0	1	0,5	1	0,5	1	Most of the guidelines clearly state their dates, however a few require additional effort to locate this information.
16	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
17	1	1	1	1	1	0	0	0	0	0	0	Around half of the GL's contribute unique insight to the research
18	1	1	1	1	1	0	0	0	0	0	0	Around half of the GL's contribute to strengthen or refuting the current position
19	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	Not deemed applicable for our context
20	0,5	0,5	1	1	0,5	0	0	0,5	1	1	1	There is an equal distribution among third, second, and first-tier articles.
Sum	12	11	12	12,5	10,5	6	9	8,5	12	11,5	11	
Normalized (0-1)	0,85	0,78	0,85	0,89	0,75	0,42	0,64	0,60	0,85	0,82	0,78	

Appendix G: Overview of which focus areas the informants reviewed

Interviewee	Secrets	Container sec	Sec testing	Artifact security	Pipeline security	Software supply chain + 3rd	Skills and awareness	Config management	IAM	Monitoring
Interviewee 1	X	X								
Interviewee 2		X	X	X						
Interviewee 3	X	X								
Interviewee 4			X		X	X	X	X		
Interviewee 5	X	X			X					
Interviewee 6	X	X	X	X	X	X	X	X	X	X
Interviewee 7	X	X	X		X	X	X		X	
Interviewee 8	X	X	X	X	X	X	X	X	X	X
Interviewee 9	X	X								
Interviewee 10	X	X								
Interviewee 11	X								X	X
Informant 12	X	X			X	X		X	X	