

MASTER THESIS  
CYBER SECURITY



RADBOUD UNIVERSITY

---

**Evaluating the Security Maturity  
of DevOps Teams**

---

*Author:*  
Onno de Gouw  
s1025613

*First supervisor/assessor:*  
dr. ir. Erik Poll  
e.poll@cs.ru.nl

*Second supervisor:*  
IT security wizard Martijn Bot  
martijn.bot@nl.abnamro.com

*Second assessor:*  
prof. dr. Eric Verheul  
e.verheul@cs.ru.nl

August 23, 2023

# Acknowledgements

I would like to express my gratitude to my supervisors dr. ir. Erik Poll, affiliated with Radboud University and IT-Security Wizard Martijn Bot, affiliated with ABN AMRO, for all their feedback and guidance during this master thesis research project. Also, I would like to thank ABN AMRO, and specifically the entire CISO, Technology and Engineering department including in particular the Application Security team (Appsody), for making my internship a very pleasant stay. Furthermore, I would like to thank dr. Charles Weir from Lancaster University for supporting me while carrying out his Team Security Assessment, as part of the Secure Development toolkit, and for providing me with the resulting reports. Additionally, my gratitude goes out to Maria Postupaeva, Juan de las Cuevas Caminero and Mohanad Alkhouli as security enthusiasts in the mobile application development teams and members of the security guild, for helping me to conduct my research with various development teams, for participating in my evaluation interview sessions, and for providing me with more insight into actual current processes at ABN AMRO. Finally, I would like to thank all participants who were open to participate in my security maturity evaluations through Weir's Team Security Assessment.

## Abstract

The number of cyberattacks targeting organizations, including banks, is increasing. Because of this, it becomes more and more important these days to be resilient against cyberattacks and mature on your security practices, in particular as a financial organization. A key factor in achieving this is by raising security maturity among the development teams within these organizations. Ideally, one should be able to systematically evaluate how mature their developers are when it comes to their security activities. Obtaining insights into which teams perform better than others and having a means to compare their security-related activities is useful, because it could provide a starting point for improving their security maturity as well. Consequently, this could enhance the security of applications, and enable quicker approval of the release of an application if teams that are more security mature are assigned to work on them.

In this thesis, we will explore existing methods to evaluate the security maturity of software development teams, and ultimately recommend an effective and practical method that a large bank like ABN AMRO could use for assessing the security maturity of such teams. We will apply our recommendation in practice with ABN AMRO's mobile application DevOps teams, validate the results and evaluate its usefulness based on evaluation interviews that we conduct with three participants who are member of the mobile application security guild at ABN AMRO. Next to this, we will attempt to bring clarity to the different scopes and confusing terminology associated with our research area, by proposing definitions and creating various scope diagrams.

In summary, we discovered many different methods to evaluate security maturity in general, such as security frameworks and security maturity models, but hardly any of them focuses specifically on evaluating the security maturity of individual DevOps teams. A different security maturity evaluation method, which uses a qualitative approach and which does focus on development teams, is Weir's Team Security Assessment. This method allows for quick evaluations and produces automated reports that facilitate useful discussions with development teams about their security maturity. Moreover, we validated the results it produces through the available test coverage security metric and based on evaluation interviews. We also clarified the different security scopes involved by creating scope diagrams, and we addressed confusing terminology by establishing a glossary containing explicit definitions. Finally, we found that ABN AMRO has numerous security-related initiatives and processes in place, and we made several noteworthy observations that could be valuable for the bank to consider.

# Contents

<b>Glossary and acronyms</b>	<b>4</b>
<b>1 Introduction</b>	<b>6</b>
1.1 Problem formulation . . . . .	7
<b>2 Software development methodologies</b>	<b>9</b>
2.1 Waterfall . . . . .	9
2.2 Agile . . . . .	9
2.3 DevOps . . . . .	10
<b>3 Incorporating security into a software development methodology</b>	<b>12</b>
3.1 Shifting security left . . . . .	12
3.2 Microsoft SDL . . . . .	13
3.3 DevSecOps . . . . .	13
3.3.1 Six Pillars of DevSecOps . . . . .	13
<b>4 Exploring security maturity evaluation methods</b>	<b>17</b>
4.1 Security framework . . . . .	17
4.1.1 Definition and relations . . . . .	19
4.2 Security maturity models . . . . .	20
4.2.1 Definition and relations . . . . .	20
4.2.2 Security maturity model scopes . . . . .	22
4.2.3 Security maturity model types . . . . .	22
4.2.3.1 Activity maturity model . . . . .	22
4.2.3.2 Capability maturity model . . . . .	23
4.2.3.3 Hybrid maturity model . . . . .	24
4.3 Weir’s Team Security Assessment . . . . .	24
<b>5 Understanding the different scopes of security maturity evaluation methods</b>	<b>26</b>
5.1 General scopes within an organization . . . . .	26
5.2 Security scopes within an organization . . . . .	27
5.3 Different scopes of security maturity evaluation methods . . .	29

5.3.1	The scope of security frameworks . . . . .	29
5.3.2	The scope of security maturity models . . . . .	29
5.3.3	The scope of Weir’s Team Security Assessment . . . . .	30
5.3.4	The scope of an ideal security maturity evaluation method focusing on DevOps teams . . . . .	32
<b>6</b>	<b>Clearing up confusing security terminology</b>	<b>33</b>
6.1	Understanding practices, activities, controls and measures . . .	33
<b>7</b>	<b>The current situation at the ABN AMRO Bank</b>	<b>36</b>
7.1	Agile development teams . . . . .	36
7.2	Apollo program . . . . .	37
7.2.1	Spikes . . . . .	38
7.3	Security initiatives . . . . .	39
7.3.1	SHARP . . . . .	39
7.3.2	Security Bites . . . . .	39
7.3.3	Security Release Checklist . . . . .	39
7.3.4	Security Champions . . . . .	40
7.3.5	Security guild . . . . .	40
7.4	CISO security processes . . . . .	40
7.4.1	DevOps Capability Assessment . . . . .	41
7.5	Security dashboards . . . . .	43
7.6	Security Strategy: Driving Secure Banking . . . . .	44
7.7	Summarizing the most important findings . . . . .	45
<b>8</b>	<b>Evaluating the security maturity of mobile application DevOps teams</b>	<b>47</b>
8.1	Applying Weir’s Team Security Assessment . . . . .	47
8.1.1	Considering four maturity evaluation factors . . . . .	48
8.1.2	Distributing Weir’s Team Security Assessment . . . . .	51
8.1.3	Results of our security maturity evaluation of mobile application DevOps teams . . . . .	52
8.1.3.1	Analyzing our results . . . . .	52
8.2	Validating our results with security metrics . . . . .	54
8.2.1	DevSecOps security metrics . . . . .	55
8.2.2	Available security test coverage data . . . . .	56
8.2.3	Comparing test coverage with our maturity results . . .	57
8.3	Evaluation interviews with mobile application DevOps teams	57
8.3.1	Validating our results with evaluation interviews . . . .	58
8.3.2	Evaluating Weir’s Team Security Assessment . . . . .	58
8.3.3	Other things that we learned . . . . .	60
8.4	Limitations . . . . .	62
8.4.1	Limitations to our security maturity evaluation process	62
8.4.2	Limitations of Weir’s Team Security Assessment . . . .	63

<b>9</b>	<b>Future work</b>	<b>64</b>
9.1	Maintaining and improving DevOps team security maturity . . . . .	64
9.2	The security maturity evaluation process . . . . .	66
9.3	Potential influences on the security maturity of DevOps teams	67
9.4	Evaluating and comparing OWASP DSOMM . . . . .	68
9.5	Internal ABN AMRO analysis . . . . .	68
9.6	Other companies . . . . .	68
<b>10</b>	<b>Conclusions</b>	<b>69</b>
10.1	Many methods to evaluate security maturity . . . . .	69
10.2	Managing Babylonian speech confusion . . . . .	70
10.3	Evaluating DevOps team security maturity using Weir’s Team Security Assessment . . . . .	71
10.4	Many ABN AMRO security processes and initiatives . . . . .	72
10.5	Recommendations for ABN AMRO . . . . .	72
	<b>Bibliography</b>	<b>75</b>
<b>A</b>	<b>Weir’s Team Security Assessment</b>	<b>85</b>
<b>B</b>	<b>Weir’s Team Security Assessment sample report</b>	<b>91</b>
<b>C</b>	<b>Weir’s Team Security Assessment results of mobile application DevOps teams</b>	<b>98</b>
<b>D</b>	<b>Template for evaluation interviews after applying Weir’s Team Security Assessment</b>	<b>107</b>

# Glossary and acronyms

## Glossary

In this Glossary we provide an overview of definitions of terms that are used in this thesis. It helps to better understand the field of our thesis and attempts to clarify confusing terminology included with this field. Throughout the chapters of this thesis, the different terms will be covered in more detail.

Term	Definition
Blocks	ABN AMRO's Agile development teams.
Checklist	A list of items used to examine if activities are completed or adhered to, involving yes or no responses.
DevOps	Development and Operations is a method that combines software development and deployment.
DevSecOps	Incorporates security within the DevOps way of working, by including security activities into each stage of their software development lifecycle.
Grids	Organizational units of several Agile development teams who take responsibility for part of the application landscape.
IT engineer	A collective term for members of Innovation & Technology in a development team at ABN AMRO.
Maturity model	"An organized way to convey a path of experience, wisdom, perfection, or acculturation, [depicting] an evolutionary progression of an attribute, characteristic, pattern, or practice" [18], typically using maturity levels.
Security activity	Any action or operation that can be taken to protect assets, individuals or information from potential security risks, threats and breaches, often supported by implemented security controls or processes, and which can be separated into SDLC related and Non-SDLC related security activities
Security control	"A [specific] safeguard, [tool] or countermeasure [put in place] for an information system or an organization, to protect the confidentiality, integrity and availability of the system and its information" [64].

Term	Definition
Security dashboard	An online visual interface providing an overview of security metrics, such as security issues or deployment frequency, and which can be filtered on the level of individual development teams.
Security framework	A set of guidelines and practices that can be used by organizations as a checklist to systematically evaluate, manage and reduce security risks.
Security guideline	Specific, prescriptive instructions or recommendations on how to practically implement security controls within an organization or system.
Security maturity	The capacity and degree of readiness to carry out security activities and follow security processes.
Security measure	See the definition of: <i>security control</i> .
Security metrics	“Quantifiable measurements used to understand the status of systems and services through the collection, analysis and reporting of relevant data” [37].
Security practice	A set of established security activities that can be followed consistently to enhance the overall security and protection of assets, individuals or information from potential risks, threats and breaches of its information and to meet a set of security requirements.
Security standard	A type of security framework that is widely accepted by organizations, governments and industry experts, defining the minimum criteria for managing and reducing security risks.

## Acronyms

In the table below, we provide an overview of acronyms that are used throughout this thesis:

Abbreviation	Meaning
CISO	Chief Information Security Officer Corporate Information Security Office
ISO	International Organization for Standardization
OWASP	Open Web Application Security Project
ISF	Information Security Forum
NIST	National Institute of Standards and Technology
PAM	Process Assessment Model
DCA	DevOps Capability Assessment
SDL	Security Development Lifecycle
SDLC	Software Development Lifecycle



# Chapter 1

## Introduction

The number of global cyberattacks that take place is increasing. According to Check Point Research, these attacks have increased by 38% in 2022 and are expected to accelerate even more with the current fast rise of AI technology, such as Chat-GPT [27]. More and more companies, such as banks, are facing cyberattacks [50]. Cybercriminals are becoming more professional and attack in various ways, like for example through bugs in the code of IT systems or injecting malicious payloads. For this reason, banks are constantly trying to elevate their security measures to a higher level. An example of this can be seen in the partnership that has been established called Partnership for Cyber Security Innovation (PCSI). In this partnership, secure development specialists from various financial companies in the Netherlands have come together to share knowledge and experience in the field of cyber security [74, 95].

One of the companies that is part of this collaboration is the Dutch bank ABN AMRO. At ABN AMRO, almost 700 development teams build and manage all kinds of different applications [1]. Two important and emerging concepts within the bank are DevOps and DevSecOps. DevOps is a software development methodology aiming to combine the work of both developers and operations teams [38], and complements the Agile methodology. DevSecOps in turn, highlights the importance of development teams working secure throughout their development practices, adding security as an integral part to the DevOps way of working. ABN AMRO is making significant investments into implementation of these new approaches, actively promoting and supporting development teams in transitioning to the DevOps software development methodology. As part of this transition, the CISO department is emphasizing the inclusion of important security considerations, to ensure a secure development process by the DevOps teams.

Naturally, different DevOps teams possess different levels of security maturity, and gaining an insight into these differences would be beneficial for the bank, or even for other organizations that develop software products,

because of various reasons. Firstly, it can help to prevent security incidents. Security mature teams are likely to build better-secured applications. If the bank knows which DevOps teams are mature and which are not, it is possible to provide these teams with targeted training and to prioritize the support provided by CISO's security teams. Moreover, it could allow for the provision of specialized resources and challenges to security mature DevOps teams, to boost their security capabilities even further. Secondly, understanding the security maturity of DevOps teams could speed up software deployment, as more mature teams may get green light for quicker application launches, simply because they make fewer mistakes and need fewer reviews.

## 1.1 Problem formulation

Security involves a lot of different aspects, and ABN AMRO has many different processes and initiatives in place already to support with and keep track of DevOps teams their security practices, as we will also see in chapter 7. Unfortunately, it remains difficult for the bank to systematically evaluate how mature ABN AMRO's development teams are when it comes to implementing security practices within the DevOps workflow. There are many different security factors involved, and they are often difficult to measure and their relation remains unclear. Moreover, the terminology used can be confusing and not well-organized. Therefore, the question is how the bank can evaluate a DevOps teams' understanding of and engagement in security practices. We formulate our research question in the following way:

**RQ:** *How to evaluate the security maturity of DevOps teams?*

In this master thesis, we will address this research question through extensive literature research and by practically conducting a security maturity evaluation with a number of mobile DevOps teams that work at ABN AMRO. Our thesis consists of ten chapters and four appendices.

In chapter 2 we will discuss different software development methodologies, such as Agile and DevOps, to get an understanding of our subject area. Next, in chapter 3 we will cover the incorporation of security into a software development methodology. Then, in chapter 4, we will discuss three methods to evaluate security maturity, which we identified based on an extensive exploration of the existing literature, and we will present two non-exhaustive overviews of them. In chapter 5 we will delve further into our findings from chapter 4, and provide a better understanding of the different security scopes covered by the security maturity evaluation methods that we identified. Chapter 6 is used to clear up and understand the confusing security terminology that we encountered during our extensive literature

research about security maturity evaluation methods and secure software development. Next, in chapter 7, we will discover the current situation and context at the ABN AMRO bank, including many of the ongoing security initiatives and processes. Then, in chapter 8 we will practically apply one of the security maturity evaluation methods that we identified with several mobile application DevOps teams at ABN AMRO, and validate those results based on available security metrics and some evaluation interview sessions. In chapter 9 we will cover any potential areas and open questions that our research has not touched upon, and which could be addressed in future research. Finally, in chapter 10 we will draw final conclusions of our research, and include some recommendations to consider for ABN AMRO.

## Chapter 2

# Software development methodologies

In this chapter, we will discuss different software development methodologies and related concepts to get a better understanding of our subject area. We start by discussing the earliest and classic linear waterfall development method in section 2.1. In section 2.2, we discuss Agile development, a different software development methodology focusing on easy adoption of change. Then, in section 2.3 we explore the latest paradigm known as the DevOps approach, which ABN AMRO is currently adopting for their development teams (see chapter 7).

### 2.1 Waterfall

A traditional software development methodology is waterfall. This method is characterized by a linear and sequential approach, consisting of different phases that must be completed before moving on to the next [34]. This makes waterfall less flexible, as software development progress flows in only one direction [104]. Even though the term waterfall was not used in the article, this methodology was first described by [79]. Because different versions of the waterfall model have been introduced over the years, these phases can vary depending on which specific model is being considered. The waterfall methodology is well-suited for small-scale software development projects with clear requirements, and feedback on the application is obtained only after the entire development and testing process is completed.

### 2.2 Agile

Agile is a different approach to software development, and a much more flexible way of working. This software methodology starts with an initial

planning phase and progresses towards the deployment phase through iterative and incremental interactions throughout a project’s lifecycle [19]. Each iteration, or sprint, can be seen as a mini-project that includes all the necessary tasks of the software development process. It has as its goal to reduce software development overhead and it allows for easy adoption of changes, collaboration between developers and customers, and continuous improvement. Additionally, it is well-suited in small and medium software development projects [19].

Agile came about in 2001 and was introduced in the Manifesto for Agile Software Development [21], when seventeen software engineering consultants attempted to more formally and explicitly define ways to handle frequent changes in software requirements and customer expectations [19, 105]. It was a shift in mindset that caused a transformation in the software development industry, enabling faster delivery of high-quality software products. The Agile Manifesto outlines four values and twelve principles that support the essence to be agile and that provide the basis to guiding the software development process [78].

## 2.3 DevOps

A very recent software development methodology is called DevOps, and it combines development (Dev) and operations (Ops) to continually produce better, more reliable products and provide value to customers [61]. After Agile gained widespread adoption for developing software, it became clear that a critical aspect was left out of scope, namely the processes and requirements of the operation teams who deployed and managed the software products. [39]. DevOps brings these two teams together, forming a single team responsible for building and maintaining developed software on the organization’s infrastructure. Agile and DevOps are complementary approaches and can coexist in an organization [97]. DevOps allows teams to build, test and release software faster, continuously and more reliably by incorporating Agile principles, practices and automated processes [96].

Since the concept of DevOps has been described in different ways and there is no standard definition for it [87], it is important to clearly define what definition we use within our research. We have decided to use the definition developed by [88]: “a set of *engineering process capabilities* supported by cultural and technological *enablers*. Capabilities define processes that an organization should be able to carry out, while the enablers allow a fluent, flexible, and efficient way of working”. Thus, according to this definition, DevOps contains three core aspects, namely capabilities, cultural enablers and technological enablers. According to [88], the enablers support the capabilities, and the capabilities include basic activities that should be carried out continuously in software engineering, namely planning, development,

testing and deployment. Furthermore, [88] states that setting up technological enablers in an organization involves selecting, configuring, and designing the right tool, while establishing cultural enablers takes time because people need to adapt to changes, and because it requires time and resources to make improvements. We will now list the capabilities, cultural enablers and technological enablers established in [87, 88] in table 2.1.

<b>Capabilities</b>	Continuous planning
	Collaborative and continuous development
	Continuous integration and testing
	Continuous release and deployment
	Continuous infrastructure monitoring and optimization
	Continuous user behavior monitoring and feedback
	Service failure recovery without delay
	Continuous measurement
<b>Cultural enablers</b>	Shared goals, definition of success, incentives
	Shared ways of working, responsibility, collective ownership
	Shared values, respect and trust
	Constant effortless communication
	Continuous experimentation and learning
<b>Technological enablers</b>	Build automation
	Test automation
	Deployment automation
	Monitoring automation
	Recovery automation
	Infrastructure automation
	Configuration management for code and infrastructure
	Metrics automation

Table 2.1: DevOps capabilities and enablers adapted from [87, 88]

The scientific literature sometimes reflects on the definition of DevOps by using 4 main principles in a so-called CAMS model: culture, automation, measurement and sharing [36, 41, 106]. We consider these principles to be embedded within the capabilities (measurement), cultural enablers (culture & sharing) and technological enablers (automation).

Note that clearly defining DevOps is not only useful for our research, but according to [87], it also helps to have a shared and clear understanding of what DevOps means within an organization. Without a shared meaning, misunderstandings, goal misalignment and missed benefits become more likely.

## Chapter 3

# Incorporating security into a software development methodology

The software development methodologies that we discussed in chapter 2 are not inherently secure. To resolve this, security practices can be implemented within the software development methodology that is used by a development team, or more general, by an entire organization. In this chapter, we discuss how security aspects are commonly incorporated into these software development methodologies, by first discussing the concept of shifting security left in section 3.1. We then discuss Microsoft's Secure Development Lifecycle in section 3.2, followed by an exploration of a method highly relevant for our research: the emerging paradigm of DevSecOps (section 3.3).

### 3.1 Shifting security left

In traditional software development methodologies, security was a step close to the end of the process [62]. However, an important principle that characterizes secure software development and was not yet present in the software development methodologies discussed in chapter 2 is *shifting security left*. It encourages to move security from the right - the end - to the left - the beginning - such that it is included in every part of the software development lifecycle [12, 62, 89]. Involving security early in the development process enables an easier integration of security controls, avoiding issues once the system is operational.

## 3.2 Microsoft SDL

Microsoft SDL, or the Security Development Lifecycle, is a framework that consists of a set of practices that support security assurance and compliance requirements which help developers to build more secure software [59]. It contains 12 practices, varying from providing training, performing static and dynamic code analysis, threat modeling and penetration testing. The aim of SDL is to minimize the severity and number of vulnerabilities in software.

These 12 security practices of Microsoft SDL can be integrated into each phase of the software development lifecycle, regardless of the methodology that is being used [53, 16]. For example, SDL can be used along with Waterfall, as discussed in section 2.1, with Agile (section 2.2) and even along with DevOps (section 2.3). In the latter case, a transition occurs towards the adoption of a new methodology called DevSecOps [49, 60]. A reason to combine SDL with the DevOps methodology could be because SDL does not include activities for operations engineers. We will discuss DevSecOps in more detail in the next section 3.3.

## 3.3 DevSecOps

An emerging paradigm that integrates security practices into the DevOps way of working (see section 2.3) is called DevSecOps. Specifically, DevSecOps has the purpose to integrate security activities into the DevOps software development lifecycle [31, 62]. It also encourages cooperation between security teams, development teams and operations teams.

### 3.3.1 Six Pillars of DevSecOps

The Cloud Security Alliance has published a series of reports defining six main principles that are essential when implementing and integrating DevSecOps within an organization [28]. These principles are called the Six Pillars of DevSecOps and aim to guide organizations and address weaknesses in secure software development within the context of DevSecOps [29]. We will now briefly discuss each pillar, while noting that these principles can be seen as an extension of the CAMS model that we highlighted in section 2.3. The Six Pillars of DevSecOps are:

- (1) *Collective Responsibility*: This principle describes that security should no longer be seen as the responsibility of someone else, or separate from business objectives, but that a new *mindset* regarding software security has to emerge within the organization. Everyone is responsible for the organization's security stance and must be aware of their own contribution.



- (2) *Collaboration and Integration*: This principle describes the importance of collaboration instead of confrontation when trying to achieve security, due to the large gap between skill (knowledge) and talent (resources) in the software landscape across Development, Operations and Security.

Clearly, pillars (1) and (2) have to do with creating a security aware and collaborative culture at an organization. This is an aspect that we often find in other literature as well [40, 60, 62, 96].

- (3) *Pragmatic Implementation*: This principle describes that organizations often struggle to find the right tools and solutions when implementing application security within their software development lifecycle, and end up with solutions that are difficult to deploy and do not effectively address security risks. Therefore, pillar 3 specifies a comprehensive approach to embedding security into the software development lifecycle and guiding organizations implementing DevSecOps to bridge the gap between development, operations, and security.
- (4) *Bridging Compliance and Development*: This principle describes that it is challenging to translate risk-related requirements and compliance needs into security and product requirements that can be measured. Therefore, this pillar tries to address this gap and guides organizations in recognizing compliance objectives, translating them to appropriate security measures, and embedding, automating, measuring and testing security controls at key points in the software development lifecycle.
- (5) *Automation*: This principle describes that automated security practices are very important for process efficiency, because they reduce manual tasks and improve testing and feedback frequency. It should be implemented wherever possible in order to improve software quality, even if challenges arise, because without automated quality checks, manual coding can result in poor performing and insecure software.
- (6) *Measure, Monitor, Report and Action*: Finally, this principle describes that actionable metrics are critical to monitor in a DevSecOps environment, because without those, progress cannot be measured and failures cannot be detected in a timely manner and DevSecOps will not succeed.

In addition to these six main principles, we can explore relevant security practices that characterize DevSecOps. Various lists of such DevSecOps security practices are available in literature and online [35, 57, 62, 76]. While these practices often overlap across different sources, they also frequently differ, making it challenging to obtain a comprehensive overview. If we consider the Cloud Security Alliance’s Six Pillars of DevSecOps, and particularly focus on pillar 3, we can identify 35 distinct DevSecOps security

activities. These activities align with the concept of shifting security left, integrating security into every stage of the software development lifecycle. Figure 3.1 illustrates the different DevSecOps security activities. Since there are many different ways to present a software development lifecycle, the 5 stages displayed here provide a universal view for software development:

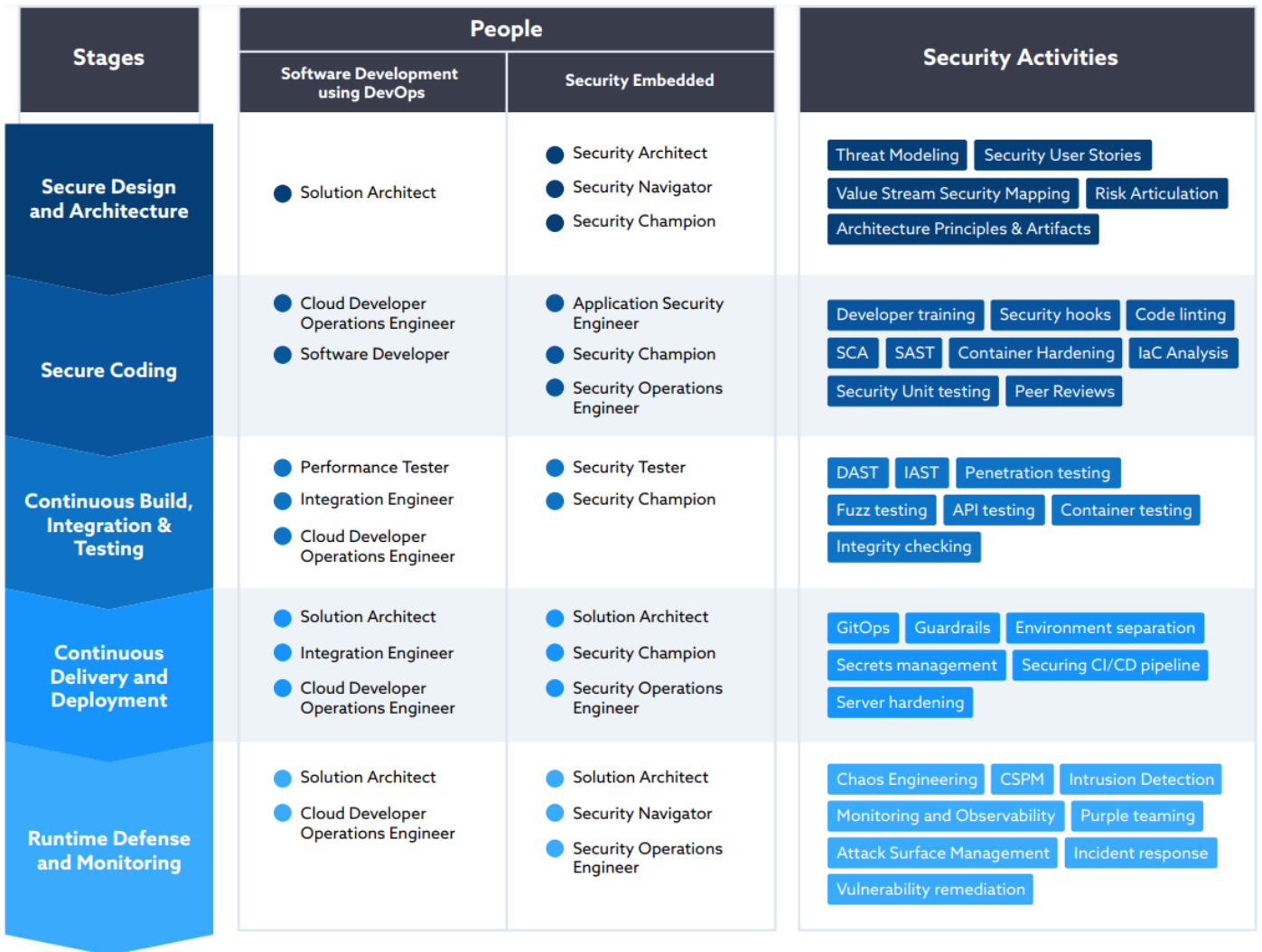


Figure 3.1: DevSecOps security activities and people in the software development lifecycle [31].

To better understand how security is integrated into the software development methodologies discussed in chapter 2, we created a graphical diagram that visualizes the relation. This diagram can be found in figure 3.2:

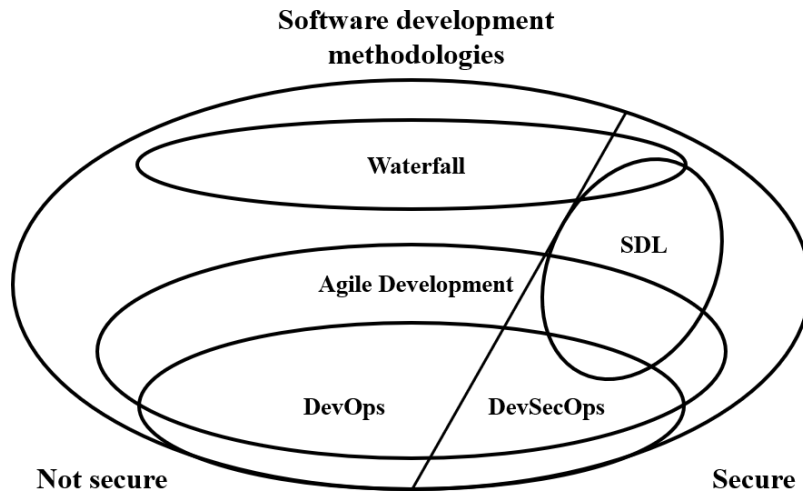


Figure 3.2: Visualization of the scopes and the relation between the different software development methodologies and the integration of security.

## Chapter 4

# Exploring security maturity evaluation methods

In literature, we encounter many different methods that can be used to evaluate security maturity. Understanding their relation and differences can be very confusing. We will use this chapter to explore the different security maturity evaluation methods that exist and we try to clear up their definitions and how they relate. We take a wide and general view, meaning that not all of these methods are specifically focused on evaluating the security maturity of DevOps teams (see section 2.3). During our exploration, we take into account several aspects that could be interesting for deciding which security maturity evaluation method can be used for DevOps teams specifically. In particular, we briefly highlight the different scopes of security maturity evaluation methods, but we will further discuss these scopes in chapter 5. To structure our exploration of security maturity evaluation methods in this chapter, we use the following division. Here, Weir's Team Security Assessment is a standalone instance that has no overarching type name, but it is not a security framework or a security maturity model.

1. Security frameworks (section 4.1)
2. Security maturity models (section 4.2)
3. Weir's Team Security Assessment (section 4.3)

### 4.1 Security framework

One of the first methods that we encountered and which could be used for evaluating security maturity is using a security framework. Many different security frameworks exist, and we provide a compact, non-exhaustive overview of the most prevalent security frameworks that we identified during our literature investigation in table 4.1:

Security frameworks	Availability	Comments	Standard
ISO/IEC 27001 [45]	Proprietary	World's best-known standard for information security management systems.	Yes
COBIT 5 [47]	Proprietary	Guidance with IT management and IT governance. Consists of different components, such as management guidelines, requirements and process descriptions.	No
CIS Critical Security Controls (CSC) [25]	Open	18 prescriptive and prioritized security best practices. Version 8 combines and consolidates the controls by activities.	No
NIST SP 800-53 [63]	Open	Catalog of security and privacy controls for information systems and organizations.	Yes, e.g. for US federal government agencies.
ISA 62443 series [46]	Proprietary	A series of standards defining best practices for security. Applicable across diverse industries.	Yes
NIST Cyber Security Framework [65]	Open	Guidelines and best practices based on all frameworks above. Categorized as identify, protect, detect, respond and recover.	No
(Defense) Cybersecurity Maturity Model Certification (CMMC) 2.0 [32]	Open	Consists of 14 cyber security domains with 110 security requirements based on NIST SP 800-171 and NIST SP 800-172. Includes maturity model characteristics, namely 3 CMMC levels.	Yes, for contractors in the Defense Industrial Base

Table 4.1: A compact, non-exhaustive overview of security frameworks.

### 4.1.1 Definition and relations

It is important to understand that security frameworks are not security maturity evaluation methods by definition. However, they are often being used as such via a *checklist approach*, in which each assessment criterion is evaluated by answering either yes or no [58]. For example, a security maturity assessor could go through the list of security practices specified in the Microsoft SDL security framework and verify whether these practices are being carried out. To highlight this *checklist approach* and because many different definitions of a security framework exist, we define a security framework as: a set of guidelines and practices that can be used by organizations as a checklist to systematically evaluate, manage and reduce security risks. This way, we aim to clarify the precise definition of a security framework.

Please note that taking the *checklist approach* for evaluating security maturity is not a preferred option, because security matters are rarely clear-cut, and we prefer to assess *how well* the assessment criterion is being used [58]. This makes using a security framework not ideal for security maturity evaluations.

Based on our security framework identification in table 4.1, we observed two things that we would like to highlight. Firstly, we noticed that various security frameworks are often based on other security frameworks. For example, the NIST Cyber Security Framework [65] is based on a subset of the security frameworks displayed in table 4.1. Secondly, we frequently encountered the term security standard while searching for security frameworks. We understand security standards as a type of security framework that is widely accepted by organizations, governments and industry experts, defining the minimum criteria for managing and reducing security risks.

To visualize the relation of these different concepts, figure 4.1 shows that security frameworks are often based on each other and security standards are a type of security framework:

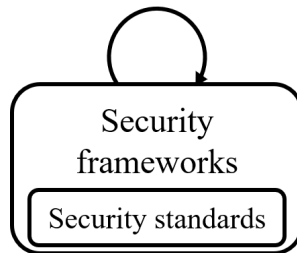


Figure 4.1: The relation between security frameworks and security standards.

## 4.2 Security maturity models

Another method to evaluate security maturity is using a security maturity model. Again, many different security maturity models exist in literature, and we considered it would be infeasible to list all of them. In table 4.2, we present a compact overview of security maturity models that we consider to be the most relevant, and we left others out of scope. Note that some of these models were also discussed in [56], who tried to do something similar.

### 4.2.1 Definition and relations

Based on our identification of security maturity models in table 4.2, we find that security maturity models make use of a *levels-scoring approach*, opposed to the *checklist approach* that is used when evaluating security maturity using security frameworks (see section 4.1). With a *levels-scoring approach*, we mean that the outcomes of a security maturity model are represented as some level or score that reflects a certain degree of security maturity. Let us consider the general definition of a maturity model: “A maturity model is an organized way to convey a path of experience, wisdom, perfection, or acculturation and depicts an evolutionary progression of an attribute, characteristic, pattern, or practice” [18]. Such a model provides a means for measuring performance and evaluate current approaches. Moreover, it generally also expresses a body of knowledge of best practices, which can help to develop capability and improve over time. Thus, maturity models define a route of progress [43], and typically this is being done through various levels that build on each other. The amount of levels and their meaning vary depending on the maturity model that is being used [52], but most maturity models include at least some non-existent or basic, intermediate and advanced level. One may think a company should always strive for the highest maturity level, but depending on the type of maturity model used it may be sufficient to develop certain processes less extensively.

Naturally, within our research we are interested in maturity models that are specifically focused on security. These security maturity models provide a way to assess security maturity and identify areas for improvement in terms of security practices, policies, activities and controls. Based on our security maturity model identification in table 4.2, we observed that generally, security maturity models are based on security frameworks, as we have seen in section 4.1. This makes it possible to measure *how well* the security practices defined in the security frameworks are being used, instead of evaluating these practices using the yes or no *checklist approach* (see section 4.1).

To visualize how these different concepts relate to each other, we created figure 4.2, which shows that security maturity models are generally based on security frameworks.

Security maturity model	Availability	Scope	Type	Comments
BSIMM [93]	Proprietary	Organization	Activity	A descriptive security maturity model. Enables comparison with other organizations across industries.
OWASP SAMM [72]	Open	Organization	Hybrid	A prescriptive security maturity model with 15 security practices containing security activities.
OWASP DSOMM [69]	Open	Secure software development	Activity	Focuses on security activities specifically used in DevSecOps implementations.
ISF Maturity Model [42]	Open	Organization	Hybrid	Enables measurement in 21 security domains. Adaptable for companies using various security standards.
NIST CSF Maturity Tool [58]	Open	Organization	Capability	Divides what should be done from what is being done, by separately assessing policy maturity and practice maturity of each NIST CSF control.
FSIMM [81]	Proprietary	Organization	Unsure	Its structure and maturity descriptions do not align with any standards [43].
C2M2 [66]	Open	Organization	Hybrid	Includes more than 350 security practices categorized into 10 domains.
CMMI Cybermaturity Platform [44]	Proprietary	Organization	Capability	Security risk assessment framework generating a maturity roadmap based on evidence.
COBIT 5 PAM [48]	Proprietary	Organization	Capability	Based on [90], it is outdated and not compliant with the newer ISO/IEC 330xx standards.
CYSFAM [73]	Open	Organization	Hybrid	Created at Utrecht/Leiden University. Complements the ISFAM model [91].
CCSMM [24, 102]	Proprietary	IT ecosystem	Capability	Focus on entire communities (cities, nations) with general recommendations not matching our research goals.

Table 4.2: Compact non-exhaustive overview of security maturity models.



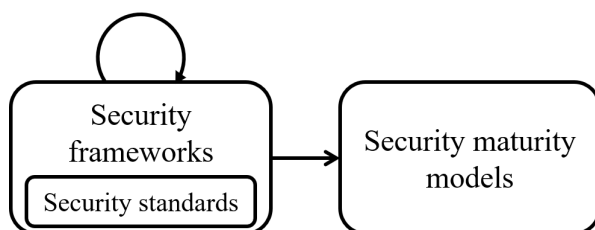


Figure 4.2: The relation between security frameworks, security standards and security maturity models.

### 4.2.2 Security maturity model scopes

During our security maturity model exploration, we observed that security maturity models contain different scopes. In table 4.2, we see that all of our identified security maturity models, except for OWASP DSOMM, aim to evaluate the security maturity of the complete *organization*, instead of the *secure software development* scope that we are focusing on with our research. Unfortunately, we identified OWASP DSOMM only at the end of our research, so we leave this model open for investigation in future work (see section 9.4). In chapter 5 we will further discuss our understanding of the different scopes involved with security maturity evaluation methods.

### 4.2.3 Security maturity model types

During our security maturity model exploration, we identified that there are three different types of maturity models with their own consistent view on progress in maturity levels [22, 23, 43]. These maturity model types are applicable for security maturity models as well. We will discuss these types in sub-sections 4.2.3.1, 4.2.3.2 and 4.2.3.3, by explaining the type of progress it measures and what to look for when choosing one as an organization. However, this classification of different security maturity model types turned out to be irrelevant for the results of our research in the end.

#### 4.2.3.1 Activity maturity model

The first maturity model is an activity maturity model. This model measures *activities* that are performed and provides a certain assurance in these activities. It represents the progression in a certain domain, such as security [23]. It is clear what the next stage of progress looks like and it is not always necessary to achieve the highest possible mature state. For example, a maturity progression for counting could be (1) pencil and paper, (2) abacus, (3) calculator, (4) computer. According to [43], these models are often seen as very practical, describing real-world activities, making it easy to assess the current maturity. However, since there is no overall scale for

the maturity model across different security disciplines, there is no way to justifiably compare maturity levels. In these, often numerical scales, one cannot state that a ‘4’ represents the same amount of progress in two different disciplines. Therefore, to know what other organizations are doing in one specific security discipline, e.g. access management, an activity model should be used [43]. Note that sometimes, ‘activity’ maturity models are referred to as ‘progression’ maturity models. We have decided to use the term ‘activity’ maturity model within this master thesis.

#### 4.2.3.2 Capability maturity model

A capability maturity model shows progress in *capability* through progress in processes. It measures a representation of organizational capability around a set of attributes, characteristics, patterns, or practices [22]. Here, capability is the extent to which the process meets current or projected business goals [43]. Since processes tend to develop in a particular way with certain characteristics, it is possible to provide a set of generic descriptions of how processes mature and apply these to each security discipline. Therefore, comparisons between processes in other security disciplines can be made. These generic descriptions can be related to the following five capability levels that are generally used, albeit in a different form or with a different name [33, 82, 103]:

- *(Level 0 - Non existent)*: At this security maturity level any security practices and processes are lacking and the company has not recognized that there are issues that need to be resolved.
- *Level 1 - Initial / Ad Hoc*: At level 1, security is usually handled in an ad-hoc way, and no formal, standardized security practices have been defined. There is a lack of security policies, procedures, and documentation, and only when an issue arises, security is addressed.
- *Level 2 - Repeatable, but Intuitive*: At level 2, security processes are documented sufficiently, and a basic security program is established using policies and procedures. This way, repeating the same procedure steps by different people is possible, but security is often still reactive.
- *Level 3 - Defined*: At security maturity level 3, the organization has well-defined and well-documented its security program. Processes and procedures are standardized organization-wide and also communicated via training and leadership. Security is being done proactively.

- *Level 4 - Managed and Measurable / Capable*: At level 4, the management monitors and measures compliance with the security procedures using established advanced security metrics and they take action if processes are not effective. The security program is continuously being improved by using the data to identify areas for improvement.
- *Level 5 - Optimized*: Finally, at maturity level 5, an organization has fully optimized its security program. Security processes are continuously analyzed and improved. Moreover, the organization has a culture of security, and security is integrated into all aspects of the organization's processes.

Increasing maturity in a capability maturity model increases the assurance that activities will be consistent, effective and resilient. However, since process descriptions are so generic and conflicts exist between how an activity theoretically matures consistently and how it matures in the real-world, accurately assessing the current maturity level becomes difficult. To compare your organization's capability across several different disciplines of security, a capability model should be used [43].

#### 4.2.3.3 Hybrid maturity model

Finally, the hybrid maturity model combines elements of both the activity and capability models by assessing progress in activities and capabilities at the same time [22, 54]. Thus, a hybrid model specifies activities that show or represent progress between levels in capability. For example, a hybrid maturity progression for security responsibilities could be (1) responsibilities are identified and assigned to people, (2) responsibilities are assigned to specific roles and documented, (3) responsibilities are reviewed, updated and managed to ensure adequacy. It increases the assurance that the organization is conducting the same activities as others, and that the organization is becoming more capable at those activities [43]. However, because hybrid models combine descriptions of activities and descriptions of how they mature for each discipline, both these descriptions are high-level. A hybrid model will only provide the high-level steps needed to increase maturity. If you want to compare your organization's high-level activities and the capability in those activities across several different disciplines of information security, a hybrid model should be used [43].

### 4.3 Weir's Team Security Assessment

After more literature research, we encountered another security maturity evaluation method, developed by Dr. Charles Weir and his team at Security Lancaster of Lancaster University, called the Team Security Assessment [86].

This free security assessment takes a different approach than the *checklist* or *levels-scoring* approaches discussed in sections 4.1 and 4.2, namely a *qualitative approach*. This is because the results generated by the assessment are qualitative instead of quantitative, and do not represent some final security maturity level or score. Weir’s Team Security Assessment consists of a practically applicable self assessment questionnaire (see appendix A) that is intended to be completed within 10 minutes by individual development teams to obtain insight in their current secure development activities. It is not a security framework or security maturity model as described in the previous sections 4.1 and 4.2.

The questions that participants have to answer depend on the role they fulfill within their development team, either technical or non-technical. After the questionnaire has been completed by more than at least 4 participants, an automated Development Team Security-Readiness Report is created. The minimum of at least 4 participant is necessary to keep individual results private. The Development Team Security-Readiness Report summarizes a team’s security awareness and their usage of a specific set of security activities, and visualizes this using four informative diagrams (see appendix B). This set of activities is limited to activities that, according to [100], are most effective and lead to better security when using the Agile software development methodology. Two of the included diagrams categorize the security activities into problem finding techniques and process improvement techniques. The Development Team Security-Readiness Report also includes a diagram visualizing the importance that team members place on the different aspects of software development, including security considerations. Finally, it includes a diagram that shows how aware a team is of different education techniques, and some suggestions about how they could improve their security-readiness.

Weir’s Team Security Assessment is part of a larger package called Secure Development [85], which contains various free resources aimed at helping software development teams with improving their security skills [26, 83]. These resources have been created and improved based on results that were gathered from highly-structured trials with several companies and based on various scientific studies [98, 99, 100].

## Chapter 5

# Understanding the different scopes of security maturity evaluation methods

In chapter 4 we identified three different security maturity evaluation methods, namely security frameworks, security maturity models and Weir’s Team Security Assessment. As we already mentioned in section 4.2.2, these evaluation methods have certain scopes. Therefore, in this chapter we try to understand the different scopes that are involved with security maturity evaluation methods. In section 5.1, we start with explaining the general scopes that exist within any organization based on the three key dimensions, namely people, process and product. Then, in section 5.2, we build on the scopes identified in section 5.1, and we focus ourselves specifically on security to find the different security scopes present within an organization. Finally, in section 5.3 we discuss the different scopes of the security maturity evaluation methods that we identified in chapter 4, based on our scope insights from section 5.2. We also consider the scope that we believe an ideal security maturity evaluation method, useful for evaluating the security maturity of DevOps teams, should cover.

### 5.1 General scopes within an organization

Within any organization, there are 3 key dimensions that can be used for a general analysis of the organization, namely people, process and product [107]. These three elements, often referred to as the three P’s, are frequently associated with management and business principles [55]. The three P’s serve as a valuable framework for understanding and evaluating the diverse activities carried out by an organization. We use them to simplify and help understand the different scopes that exist within an organization as displayed in figure 5.1.

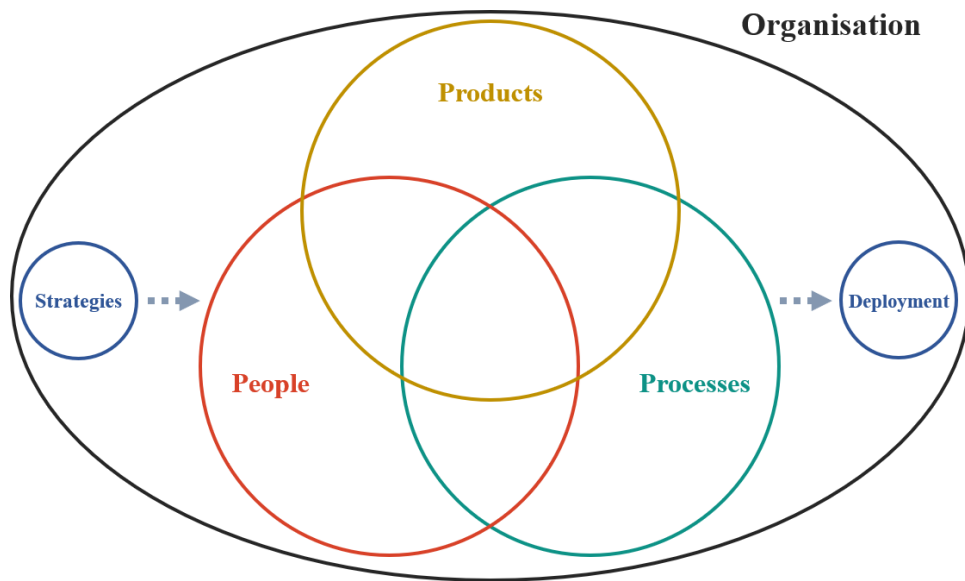


Figure 5.1: The general scopes that exist within an organization.

Note that figure 5.1 also includes the different strategies that organizations often use to steer their business towards certain objectives. Additionally, we included the deployment of products, plans or ideas by organizations.

## 5.2 Security scopes within an organization

Based on our understanding of the different scopes that exist within an organization and the three P's framework, we are also able to specifically analyze security within an organization. By focusing ourselves on security, we came to the identification of scopes as presented in figure 5.2, with the more specific dimensions of DevOps teams, security processes and software products. We also provided names to the different visible regions, to help understand what the different security scopes entail.

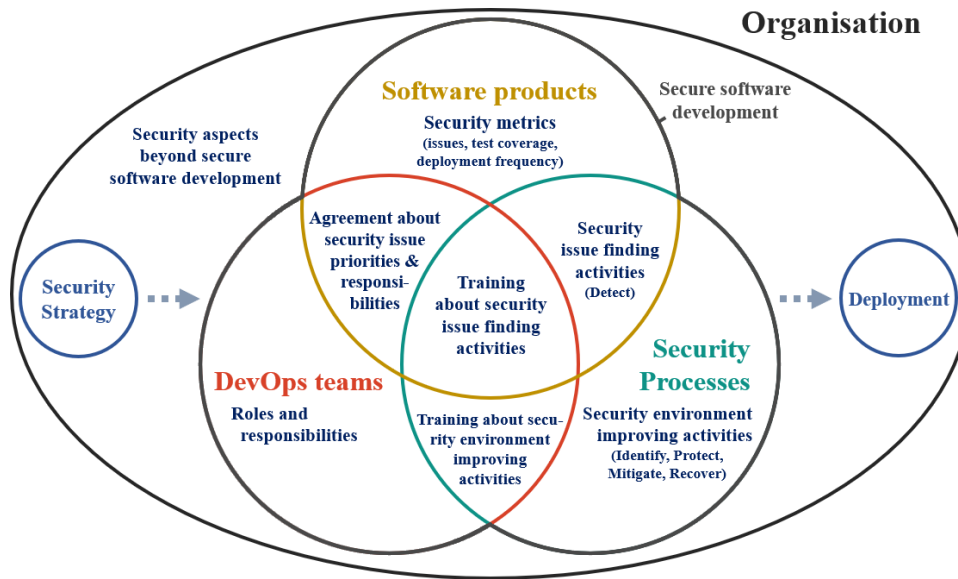


Figure 5.2: The different security scopes within an organization.

Figure 5.2 shows that if we focus ourselves on security, we can consider the combination of DevOps teams, security processes and software products as being the scope of secure software development, which is what our research focuses on. Beyond this secure software development scope, there are other security aspects that exist within an organization, such as a security strategy and the deployment of software applications. We will now provide a brief overview of the different areas that we identified within the scope of secure software development (see figure 5.2).

- *Security issue finding activities*: These are security activities closely related to the software product, as they aim to detect security vulnerabilities within the software. An example would be a penetration testing process.
- *Security environment improving activities*: These security activities aim to improve the security environment of the organization, by avoiding threats via early identification and protection measures. They also include processes for mitigating and recovering from security incidents. Examples would be threat modeling and vulnerability remediation.
- *Training about security issue finding and security environment improving activities*: As the name suggests, these two areas specify that training is important to ensure that teams know what security activities they should carry out, how and why they should do this.

- *Security metrics*: These are “quantifiable measurements used to understand the status of systems and services through the collection, analysis and reporting of relevant data” [37]. Typically, they are directly related to the software product. Examples include security issues, security test coverage and deployment frequency.
- *Agreement about security issue priorities & responsibilities*: We use this label to represent the overlapping region between software products and DevOps teams. It is important for teams to align on how they prioritize security issues in applications, and to define clear responsibilities for different security matters.
- *Roles and responsibilities*: These represent the current roles with associated responsibilities that each DevOps team already has for their secure software development practices.

### 5.3 Different scopes of security maturity evaluation methods

Based on our security scope insights discussed in section 5.2, we are able to determine the scopes that correspond with the different security maturity evaluation methods that we identified in chapter 4.

#### 5.3.1 The scope of security frameworks

During our identification of security frameworks (see section 4.1) we did not consider the different security scopes of these frameworks. This is because the *checklist approach* that is generally being used if one decides to evaluate security maturity using a security framework, does not provide insight into *how well* the different assessment criterion are being used and because security matters are rarely clear-cut [58], as explained in section 4.1.1.

However, we expect that the security scope of a security framework will differ depending on the security framework one considers. Often, the scope will probably be the same as that of security maturity models (see section 5.3.2), given that security maturity models are generally based on security frameworks. However, some security frameworks, such as Microsoft SDL, do entail the secure software development scope.

#### 5.3.2 The scope of security maturity models

As mentioned in section 4.2.2, if we examine our identification of security maturity models in table 4.2, we observe that all of these models, except for one, focus on the evaluation of the security maturity of the complete



organization. Therefore, these models cannot be applied for a security maturity evaluation of individual DevOps teams. This finding is also supported by [17], which states that “to date, little research has been conducted on developing models and strategies for DevSecOps practices in the real-world industry. Hence, despite the importance of incorporating security into DevOps, no maturity model supports DevSecOps in software development processes by highlighting critical success factors, critical challenges, and best practices, and a road map”. Only DSOMM focuses on DevSecOps and the secure software development scope that we are interested in, but since we only discovered it later during our research, we leave it open to future work (see section 9.4).

Of course, a security maturity model could be helpful for a CISO department that would like to evaluate the security maturity of the entire organization. We visualize the security scope of security maturity models in figure 5.3:

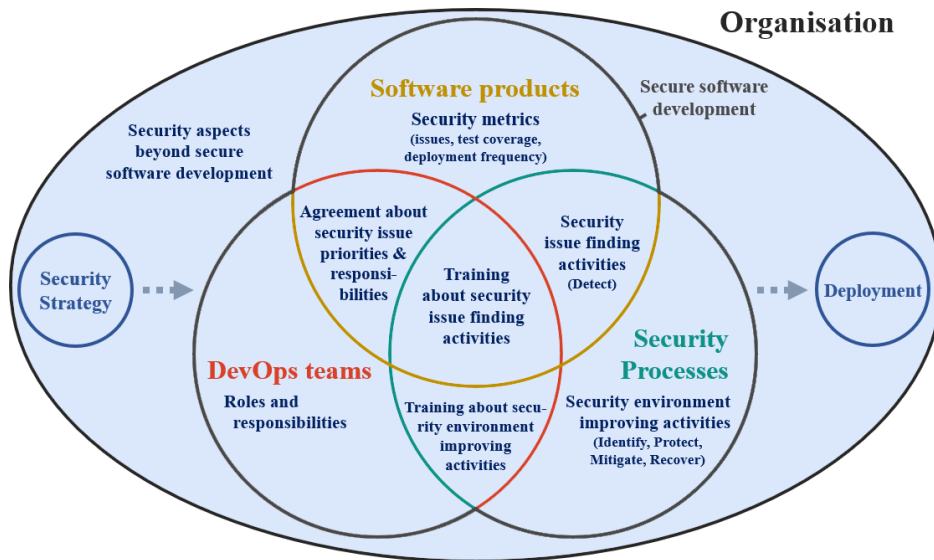


Figure 5.3: The scope of security maturity models.

### 5.3.3 The scope of Weir’s Team Security Assessment

In section 4.2, we already highlighted that almost all security maturity models take an organization-wide approach to measuring security maturity and that they are not suitable for evaluating the security maturity of individual DevOps teams specifically. If we analyze our last security maturity evaluation method, Weir’s Team Security Assessment [86], we notice that it does focus on the scope of secure software development. Specifically, Weir’s Team Security Assessment focuses on security maturity of developers following the Agile development methodology (see section 2.2), as visualized in figure 5.4.

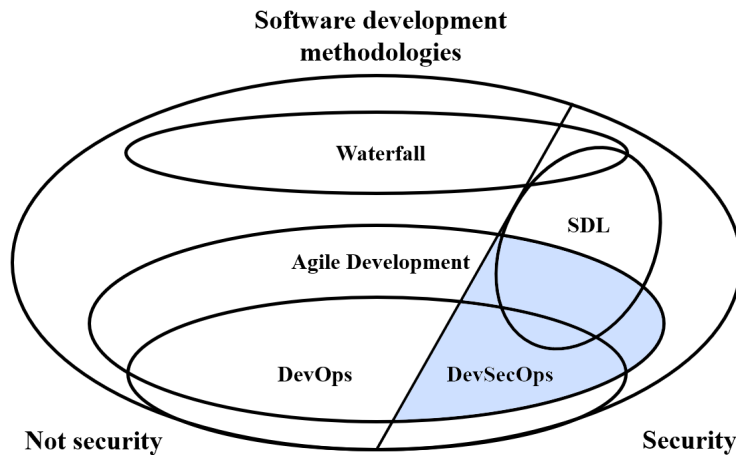


Figure 5.4: The software development methodology scope of Weir’s Team Security Assessment.

Additionally, if we apply our security scope insights discussed in section 5.2 to determine the scope of Weir’s Team Security Assessment, we notice that this assessment includes some evaluation of both security issue finding and security environment improving activities, some brief evaluation of training activities, and an evaluation of the agreement on security priority within teams. It also incorporates a different set of questions based on the role an IT engineer has within the development team, as can be seen in appendix A. The only aspect that is not being considered by Weir’s Team Security Assessment are security metrics. Based on this analysis, we visualize the scope of Weir’s Team Security Assessment in figure 5.5:

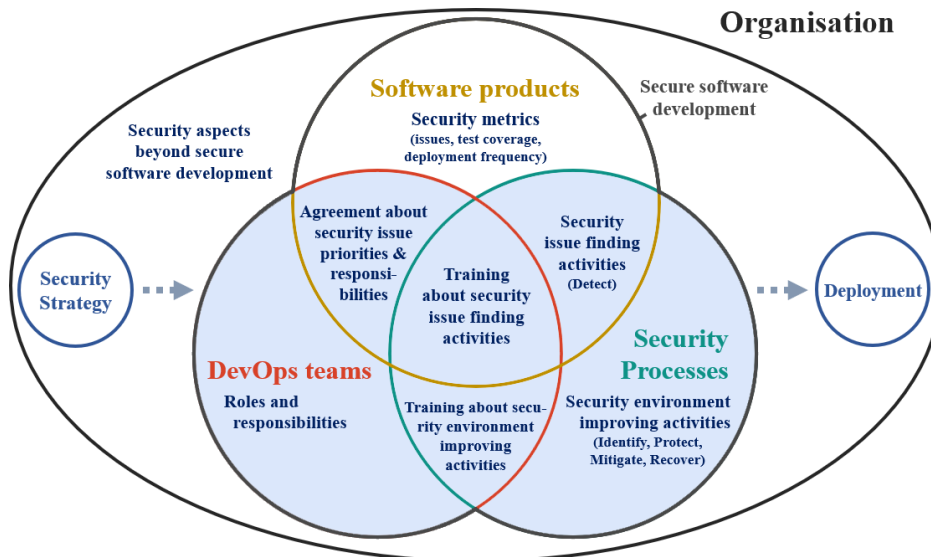


Figure 5.5: The scope of Weir’s Team Security Assessment.

From this understanding we can conclude that Weir’s Team Security Assessment incorporates almost all aspects that we would like to incorporate when carrying out a security maturity evaluation of DevOps teams. Only security metrics are left out of scope. Moreover, since Agile and DevOps are complementary approaches (see section 2.3), we do not expect this difference in software development methodology scope to be a large problem. Therefore, practically applying Weir’s Team Security Assessment with actual DevOps teams shows potential, and we have decided to do this in section 8.1.

### 5.3.4 The scope of an ideal security maturity evaluation method focusing on DevOps teams

Now that we have some insight into the different security scopes, as discussed in section 5.2, we are also able to reason about the scope of an ideal security maturity evaluation method for evaluating the security maturity of DevOps teams. Naturally, such ideal security maturity evaluation method should involve all aspects that we identified within the scope of secure software development, including security metrics, as visualized in figure 5.6.

Combining security metrics on a team level with results that Weir’s Team Security Assessment [86] produces is hard. The automatically generated reports contain qualitative diagrams, and do not contain some security maturity score that we could influence based on the quantitative security metrics. For example, we cannot lower a team’s overall security maturity score if their security test coverage is low. However, we do believe in the possibility to use security metrics to empirically validate the security maturity evaluation results found, and to provide a level of confidence in these results.

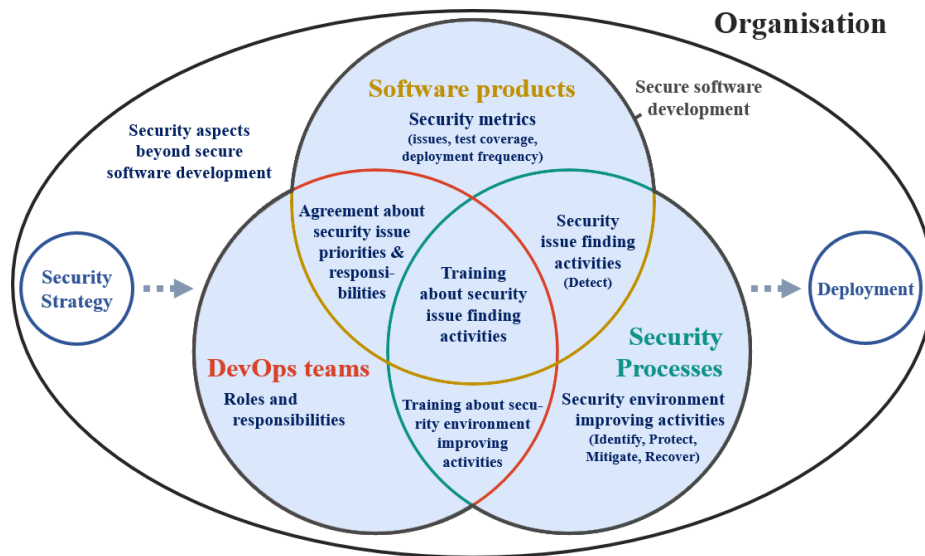


Figure 5.6: The scope of an ideal security maturity evaluation method.

## Chapter 6

# Clearing up confusing security terminology

During our identification of security evaluation methods and their scopes in chapters 4 and 5, we encountered many different security terms that were used interchangeably. There is a lot of confusing security terminology in the field, making it hard to understand the differences and similarities between different terms, and how they relate to each other. To provide more clarity, we created a Glossary that contains quite some confusing security terminology that we encountered during our research, accompanied with a definition in the way that we understand these terms now. This way, we attempt to create clarity and provide a better understanding of the confusing terminology that is being used within the security maturity literature.

### 6.1 Understanding practices, activities, controls and measures

Next to the different terms that we presented in our Glossary, there are some specific terms that we would like to highlight from this Glossary. These terms are security practices, security activities, security controls and security measures. The reason for this is because these terms are used interchangeably a lot in security literature, especially in security frameworks and security maturity models, but their precise definitions remain unclear. For example, [31] considers penetration testing a security activity, while [25] refers to it as a security control. We believe that these four terms are related to each other but are not all the same, and therefore we think it is important to work out this relation and properly define what the terms mean.

To illustrate our understanding of these different terms, we created figure 6.1, which shows the scope of each of the four security terms that we are discussing and how they are related. We have used various colors to illustrate how specific security controls, activities, and practices are associated with each other. However, it is important to note that this is only an illustrative example intended to provide insight, and no strict classification is meant. For example, the firewall security control could also be associated with activities other than environment separation, such as container hardening.

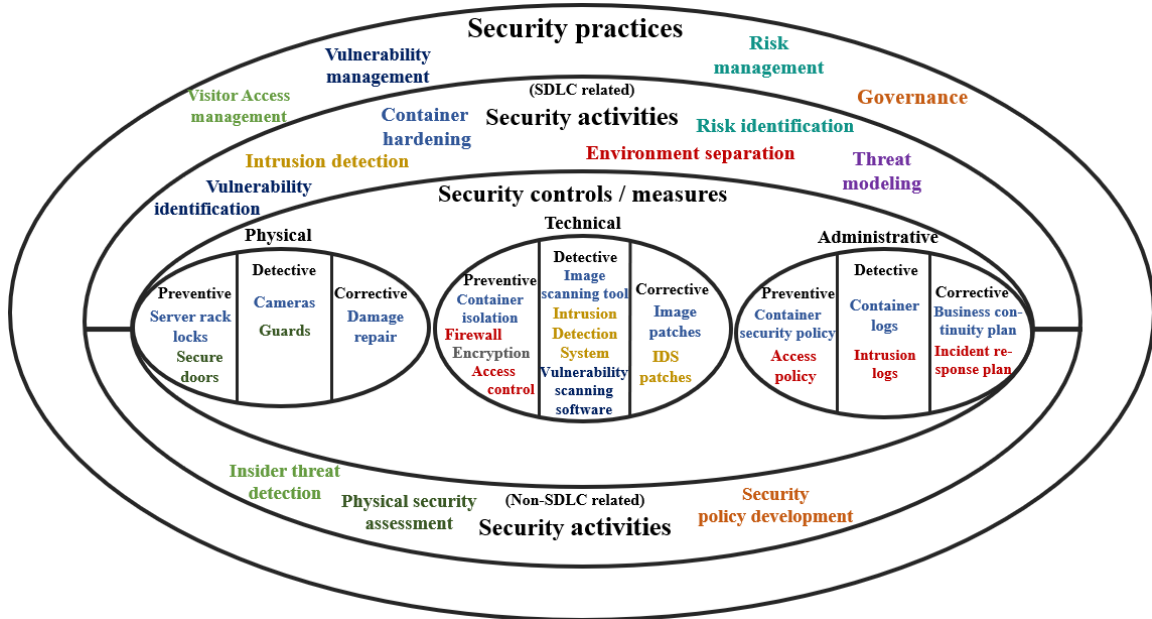


Figure 6.1: The scope of security practices, activities, controls and measures.

Basically, we regard security controls and security measures as being the same things, and the terms can be used as synonyms. They are specific safeguards, and can be divided into physical, technical and administrative security controls [92]. Here, physical controls are measures such as locks and cameras that physically restrict access to rooms and buildings. Technical controls involve using technology, such as encryption or container image scanning tools, and administrative controls include policies and logs that guide employees secure behavior to maintain a secure environment.

Every one of these security control categories can be preventive, detective, or corrective, indicating the different functions that the security controls can have [92]. Preventive controls aim to prevent vulnerabilities and threats from occurring in the first place, such as firewalls and access control. Detective controls focus on identifying potential security issues, like intrusion detection systems. Finally, corrective controls involve plans and software patches to rectify and mitigate the impact of security incidents.

To provide a clear definition of security controls, we base ourselves on the definition of NIST, which defines them as follows: “A [specific] safeguard, [tool] or countermeasure [put in place] for an information system or an organization, to protect the confidentiality, integrity and availability of the system and its information” [64].

When considering security activities, we understand these as being actions related to security which developers can carry out, and which are often supported by the security controls that are implemented. Moreover, we suggest dividing security activities into those that relate to the Software Development Life Cycle (SDLC) and those that do not. For example, environment separation would be an SDLC related activity, which developers can carry out during the deployment stage of their software development lifecycle. On the other hand, physical security assessment would be a security activity as well, but cannot be attributed to the SDLC of developers.

Based on these insights, we propose the following definition of a security activity: *Any action or operation that can be taken to protect assets, individuals or information from potential security risks, threats and breaches, often supported by implemented security controls or processes, and which can be separated into SDLC related and Non-SDLC related security activities.*

Finally, we often encountered the term of security practices during our exploration of security maturity evaluation methods in section 4. Based on our exploration, we understand a security practice as being a more global, all-encompassing term that includes a set of the security terms we discussed before. This means that security activities are individual building blocks contributing to implementation of a broader security practice. For example, vulnerability management can contain security activities such as vulnerability identification and penetration testing.

We propose the following definition of a security practice: *A set of established security activities that can be followed consistently to enhance the overall security and protection of assets, individuals or information from potential risks, threats and breaches of its information and to meet a set of security requirements.*

Even though we tried to create clarity by precisely defining security practices, activities, controls and measures, it remains important to note that the line between these terms is sometimes blurred and not as clear-cut.

## Chapter 7

# The current situation at the ABN AMRO Bank

In this chapter we try to understand and describe the large number of roles, processes and initiatives that already exist in the current situation at the ABN AMRO Bank. This provides us with background information about what the bank is already doing, and it can be helpful to understand the organization when carrying out our practical research in chapter 8. Moreover, gaining a clear understanding of the current situation at ABN AMRO could also result in a recognition of specific noteworthy observations.

First in section 7.1, we will discuss how ABN AMRO's development teams are structured and what roles are included within such a team. Then, in section 7.2, we will describe the Apollo program, which includes a migration of the development teams at ABN AMRO to the new DevOps software development methodology (see section 2.3). In section 7.3, we will elaborate on current initiatives in place at the bank and discuss their relation with improving the secure development activities of developers. After this, section 7.4 will highlight the different processes CISO carries out, while specifically focusing on the DevOps Capability Assessment that development teams can request. Then, we will discuss CISO's security dashboards in section 7.5, and we will also address the new ABN AMRO security strategy that will be in place from 2023 to 2025 in section 7.6. Finally, in section 7.7, we will summarize our observations and understanding of the current situation at ABN AMRO based on our scope insights from section 5.2.

### 7.1 Agile development teams

Agile development teams, or 'blocks' in ABN AMRO terminology, are structured in a similar fashion across the entire bank [1]. Each team contains a product owner who is responsible for prioritizing tasks and determining the product goal by specifying the features that need to be added or removed

from the application. While not necessarily technical and often from the business side, the product owner plays a crucial role in determining the direction of development. Developers and testers, on the other hand, are responsible for creating and testing new features and are part of I&T, Innovation & Technology. The number of developers that are part of a team varies with some having 3 or 4 and others having more than 16.

Sometimes, a team also contains a scrum master, who is present to provide support and remove any obstacles that may arise. They also apply Agile project management during a project. Additionally, the IT lead manages developers and testers, along with the HR aspects of the team.

A commonly used collective term for members part of I&T in a development team is IT engineer. Each IT engineer can choose a specific specialization called a ‘spike’, and we will elaborate on spikes in section 7.2.1.

Generally, each development team owns one or multiple applications. Teams are organized into grids, which are organizational units of several Agile development teams who take responsibility for part of the application landscape. For example, the Digital Channels Personal Banking grid works on creating new digital capabilities and development of mobile native apps [5].

Overall, ABN AMRO’s Agile development teams are well-structured, with each member having a defined role and expertise in a specific area.

## 7.2 Apollo program

Currently, ABN AMRO has a program running called Apollo, with the main goal to transform and simplify IT processes within the bank. This program started in 2019 and contains various aspects. For our research, the most relevant aspect of the Apollo program is the migration of the current development methodology that ABN AMRO’s developers are using to the newer DevOps methodology (see section 2.3). This migration is often referred to as an Apollo leap. As part of the Apollo leaping process, CISO’s DevOps Security team supports ABN AMRO’s development teams with the incorporation of security into their DevOps way of working, an extension also known as DevSecOps (see section 3.3).

During our research at ABN AMRO, we did two observations related to the Apollo program that might be worth considering and which we would like to mention now. Firstly, the Apollo leaping process seems to be a process that is only carried out once by development teams. However, since these teams change composition from time to time, with new members joining the bank and other members leaving, it is worth questioning what remains of the learned DevSecOps practices as time progresses. A consistent check on the security maturity of the relevant teams is not in place, as we already mentioned in section 1.1.



Secondly, we noticed that ABN AMRO has no mature definition in place of what DevSecOps entails within the organization. Because of this, it might be unclear for development teams what is expected from them when it comes to their security activities. This is also something we found based on our evaluation interviews that we will discuss in section 8.3.3.

### 7.2.1 Spikes

As part of the Apollo program, ABN AMRO introduced the concept of spikes to ensure some structured division of specialization roles within a DevOps team. Basically, a spike is an area of expertise that IT engineers and leads need to choose, next to some mandatory basic introduction into IT. The intention behind this approach is to form diverse teams capable of implementing the DevOps methodology by combining IT professionals with different spikes. There are six available spikes developers can choose from:

- *Development*: This spike focuses on development and is more than just programming. Its skill set accumulates to a full-stack developer.
- *Solution Design*: Translating requirements into platform design solutions. Having this knowledge in a team increases autonomy, as each team is owner of their own applications architectures.
- *OPS / Infra*: Includes all operations skills of code and infra required, such as automation of platform infrastructure, code and stacks.
- *Test / QA*: Concerns setup and maintenance of a desired level of quality in terms of code, infra and automation based on reviews, validation and testing.
- *Data*: This spike is all about skills required for data engineering, such as collecting, receiving, storing and delivering data to consumers.
- *Security*: Finally, the security spike includes a skill set of security practices every team ideally should have in-house as they are becoming increasingly autonomous. It is not the same as being a Security Champion, discussed in section 7.3.4.

Ideally, each development team includes at least one member who has chosen for the security spike. However, based on conversations with people at CISO, we notice that only a small number of developers choose the security spike, and not every team has members with expertise in security. Based on our research with the mobile application DevOps teams (see chapter 8) we also notice that grids, organizational units of Agile development teams, have the freedom to alter the concept of spikes, as they use the term ‘guild’ instead. In section 7.3.5, we dive deeper into the security guild.

## 7.3 Security initiatives

During our analysis of the current situation at ABN AMRO, we discovered many different security initiatives. These initiatives ranged from large-scale security programs aiming to improve the security awareness of employees across the entire bank, to smaller security training and awareness tools focusing specifically on secure software development and CISO's security processes (see section 7.4). Most of these initiatives come from the Security Culture & Transformation team, which is part of CISO's Technology & Engineering department. This team aims to build a strong security culture within the bank via awareness and education, and by supporting automation and transformation of security processes and tools [10]. It can be subdivided into three teams, namely Security Process & Tooling, Business Development & Innovation and Mindset, Awareness & Education. Our thesis is likely to be most relevant to the first and last of these three teams. We will now highlight the most important initiatives that we encountered.

### 7.3.1 SHARP

The first initiative is the awareness tool used across the entire bank called SHARP [11]. SHARP provides a continuous learning program and aims to keep all employees sharp on non-financial risks, including security risks, by answering various quiz questions correctly. ABN AMRO employees are required to maintain a SHARP score above 70%, but this score decreases over time because annual training is not enough to keep up with the fast evolution of cybercrime. Although SHARP does help employees to gain a basic understanding of cyberrisks, it does not contribute to improving the secure development activities of DevOps teams.

### 7.3.2 Security Bites

Another initiative at ABN AMRO is known as Security Bites. Security Bites are short educational videos explaining various security topics, whether directly related to the software product or not, such as the risks of hard-coded secrets and threat modeling [8]. Specifically, Security Bites are focused on helping development teams with their secure development activities. They present security knowledge, including knowledge about CISO's security processes (see section 7.4), in a visually and more comprehensible way, allowing for easier digestion of the information.

### 7.3.3 Security Release Checklist

The Security Release Checklist is a comprehensive list with many different kinds of security related activities that a development team can consider before deploying their application to production. This way, developers have

some guidelines to ensure that security activities and measures are taken into account throughout the entire software development lifecycle, and helps applying the DevSecOps approach. The initiative is still new, but could be the start of a solution to our observation that ABN AMRO does not have a mature definition of what DevSecOps entails (see sections 7.2 and 8.3).

### 7.3.4 Security Champions

The next initiative we consider is Security Champions. Security Champions is a bank-wide movement in which any employee can decide to join an open community of security enthusiasts called security champions [9]. However, generally, a security champion is being nominated during an Apollo leap of a development team (see section 7.2). They are essential for the transformation to and the establishment of a security culture. They also know when to communicate with CISO, or act as a contact point for security matters within their own team. Security champions help to make security decisions and feel responsible for the security of a product. They also have access to educational opportunities, even if it is not part of their job profile.

However, at ABN AMRO being a security champion means something different than finishing the security spike. A spike refers to a role within a DevOps team, as discussed in section 7.2.1, whereas being a security champion is not a formal IT role, but more a bank-wide community. Anyone who wants to join the community can do so, while the security spike is only available for IT engineers. An observation we make here is that, because it is not considered a formal role, the bank does not keep a formal register of their security champions.

### 7.3.5 Security guild

Finally, we consider a different kind of security initiative that we encountered, known as the security guild. This bottom-up initiative was set up by the mobile application development teams and functions as a way to organize the teams. It is their customized variation of the security spike that is used by other ABN AMRO development teams (see section 7.2.1). Similar to spikes, other guild types also exist within the mobile application development grid, but these guilds allow for flexible role changes rather than being restricted to one role. Inside the security guild, a group of mobile application security enthusiasts collaborates and shares their knowledge.

## 7.4 CISO security processes

Next to these security initiatives, we came across many processes that exist within the CISO department. These processes are designed to ensure security and to protect ABN AMRO from potential threats, by supporting

DevOps teams with their security needs. Important to note is that development teams hold the main responsibility for the security of their applications, as part of the DevSecOps implementation (see sections 3.3 and 7.2).

A quick overview of some CISO security processes available for DevOps teams include the Security Check, Security Evaluation and Advice (SEA), DevOps Capability Assessment, CIA Assessment, Penetration test, Crown Jewel Analysis, Hardening check, Application Security Monitoring (ASM) and Threat modeling training. Given this large number of processes, it is worth questioning if ABN AMRO has set up any processes to evaluate DevOps team security maturity already, and whether they are effective. We identified one process, the DevOps Capability Assessment, that might be relevant for this purpose, and we take a closer look at it now.

### 7.4.1 DevOps Capability Assessment

The DevOps Capability Assessment (DCA) is used both to guide and to evaluate development teams transitioning to the DevOps way of working, as part of the Apollo program (see section 7.2). Generally, the DCA focuses on the DevOps capabilities (see section 2.3), but it includes attention for security too. It can be used by teams to discover which DevOps and security controls they currently have in place, and which ones they still need to implement to fully adopt the ABN AMRO DevOps way of working. During the evaluation process of teams, they are required to provide evidence of their implementation of the different DevOps and security controls [2].

In short, the process is as follows:

1. A development team requests a DevOps Capability Assessment [7].
2. The team implements the DevOps controls while gathering information and arranging support if necessary [2].
3. The team submits evidence that they implemented the controls [77].
4. The team requests a review of capability supporters [13].
5. After the review and approval, teams are registered as being DevOps Control Compliant [14].

If we conduct a more detailed analysis of the DevOps Capability assessment, we can make several interesting observations.

Firstly, since a security component is included in the DCA, this suggests that it assesses not only a team's DevOps way of working, but actually also their DevSecOps practices. Because of this, we visualize our understanding of the scope of the DCA in figure 7.1 as follows:

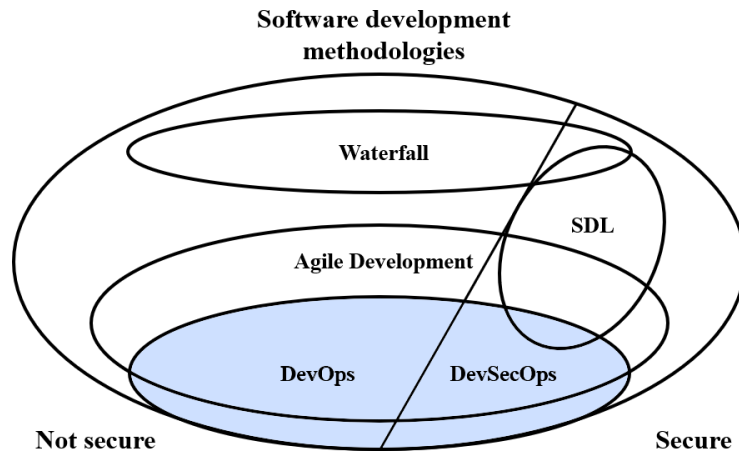


Figure 7.1: Scope of the DevOps Capability Assessment in relation to different software development methodologies and security

This scope is broader than the scope an evaluation method specifically made for evaluating security maturity would have, but it does indicate the possibility that the DCA includes some kind of evaluation of the security maturity of DevOps teams already, given that it also covers the DevSecOps approach. However, further analysis of the DCA’s security component reveals that the different security controls are described very generally, without explicitly defined DevSecOps security activities (see our identified DevSecOps security activities in section 3.3.1). For example we observe that the DCA states: “A continuous integration / continuous delivery (CI/CD) pipeline is used for secure software development and delivery throughout the Development, Test, Acceptance, and Production (DTAP) environments” [3]. We believe that stating activities explicitly instead, such as security unit testing, code reviewing, penetration testing and container testing, could allow for easier adoption by DevOps teams. Moreover, we also notice that in the DCA excel sheet, containing all the assessment criteria, the URLs for accessing reference materials related to each DevOps control are no longer functional, making them ineffective for teams seeking additional resources.

Secondly, we notice that when the DCA is used to evaluate development teams, the non-preferred *checklist approach* for the security evaluation is used, like we discussed in section 4.1.1. Therefore, it does not provide an evaluation of *how well* security controls are being used.

Thirdly, while the Apollo program recommends DevOps teams to complete the DCA on a quarterly basis [6], various internal conversations we had indicate that teams generally complete it only once, namely during their leap. Consequently, teams are registered as being DevOps Control Compliant, as mentioned in step 5 of our process description, bypassing the need to

demonstrate their implementation of DevOps and security controls through the DCA in the future again. As also seen in section 7.2, this overlooks the dynamic nature of teams, with members joining and leaving the bank. A consistent check on DevOps security maturity is not in place.

In summary, we believe that ABN AMRO's current DevOps Capability Assessment process is not ideal for evaluating security maturity of DevOps teams. It has a broader scope beyond DevSecOps, and there are several areas for improvement, such as a revision of the general descriptions of security controls included, updates of the provided URLs in the DCA excel sheet and regular intervals for DCA evaluations.

## 7.5 Security dashboards

Next to these CISO initiatives and processes, ABN AMRO's CISO also maintains various security dashboards that provide an overview of different kinds of security metrics, such as security issues, unit test coverage or dependency violations, that inform about the security status of applications via Microsoft Power BI. These dashboards can be filtered on different levels, such as on the level of development teams, but they can also provide a high-level overview of the security quality of ABN AMRO and which applications possibly need improvements.

One of CISO's security dashboards that seems to be promising for our research is the Development Analytics Dashboard. This dashboard offers insight into the code quality of ABN AMRO's applications, based on the tools Fortify, Nexus Lifecycle and SonarQube. However, if we take a closer look at this dashboard, some interesting observations can be made.

Firstly, we believe the Development Analytics Dashboard does not contain clear and complete code quality information for each of ABN AMRO's applications. Not all of the development teams are shown in the dashboard, and some teams only have a few of the applications they own and maintain visible in the dashboard. Additionally, the applications linked to each development team are presented with vague and abbreviated names, making it very unclear what they mean. Consequently, the reliability of this data becomes questionable. For example, it is unclear whether the number of open security issues displayed in the Development Analytics Dashboard for a specific team includes all issues of all applications owned by that team.

For our research in chapter 8, we focus ourselves on the mobile application DevOps teams. If we consult the Development Analytics Dashboard for these teams, we quickly realize that the observations we mentioned earlier hold for these teams as well, meaning that this dashboard does not provide us with much insight. Another dashboard that does show potential for this purpose is the Mobile DevOps Pipelines Dashboard maintained by the

mobile application developers themselves. This dashboard provides us with similar security metrics and filtering functionality as the Development Analytics Dashboard, but these security metrics seem to be clear and complete. The dashboard also seems to be more extensive. Therefore, we will use the Mobile DevOps Pipelines Dashboard for our research in section 8.2.

## 7.6 Security Strategy: Driving Secure Banking

On a special Technology & Engineering department day during our research period, a new strategy for carrying out security was presented by Global CISO Martijn Dekker and Head of CISO Technology & Engineering Rob Havermans. The strategy is called ‘Driving Secure Banking’ and focuses on a security foundation with three closely related aspects: *business resilience*, *secure choices* and *data driven decisions* (see figure 7.2). By focusing on these building blocks, ABN AMRO wants to be a safe and secure bank.

According to Martijn Dekker, *business resilience* is a key objective to achieve, as it takes a lot of time to recover from a large scale ransomware attack in which none of the IT infrastructure works anymore, and it is important for ABN AMRO to be able to keep serving customers.

The other security strategy aspects are also important. The *security foundation* is necessary to provide secure and safe services to customers of the bank. It refers to having basic security practices in place, and its operation is enabled via *business resilience*, *secure choices* and *data driven decisions*.

*Secure choices* is about awareness and making sure that other people in ABN AMRO can easily adopt and use security practices. It is about facilitating secure behavior by making secure practices more accessible.

*Data driven decisions* focuses on up-to-date data insights and using dashboards that can be used as sensors to show people where they are now and what they should be doing to improve.

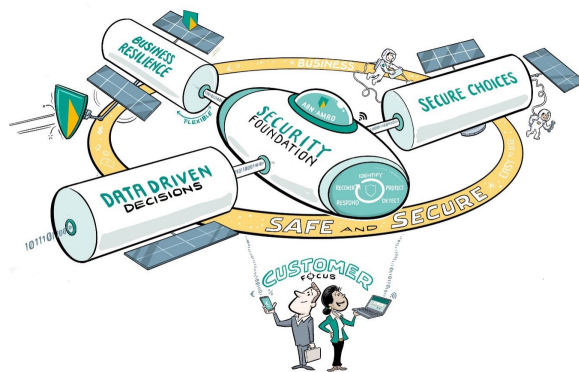


Figure 7.2: ABN AMRO’s security strategy for 2023-2025 [4].

This discussion about the new security strategy of ABN AMRO was not only to provide extra background information to our research setting, but also because our research aligns well with the new strategy. In particular, our research tries to contribute to up-to-date data insights about the security maturity of DevOps teams, which in turn can be used to make *data driven decisions* and show people where they are now and what they possibly can improve. This way, it allows DevOps teams to make *secure choices*, such that the *business resilience* of the bank will increase in the end.

Finally, during a Q&A session about the new security strategy, Global CISO Martijn Dekker explained that ABN AMRO employees, such as developers, are currently unaware of where they stand when it comes to security. Therefore, some framework or method that provides an overview of where these employees and where the organization currently stands would be desirable by the bank. Again, this clearly matches our research goals. Perhaps, one of the security maturity models discussed in section 4.2 provides a way to evaluate where the entire organization currently stands.

## 7.7 Summarizing the most important findings

In this chapter, we discussed many different aspects of the current situation at ABN AMRO Bank, and we made various observations during our research. To create a concise story, we will now summarize our most important findings in this final section.

Firstly, in section 7.2, we learned that currently, ABN AMRO has no clear and mature definition in place of what DevSecOps entails within the organization. This could cause confusion about what specific security activities are expected to be carried out by development teams, and is also a result that we found in section 8.3.3.

Secondly, from sections 7.3 and 7.4 we learned that ABN AMRO, and more specifically its CISO department, has a large number of initiatives and processes that focus on security and secure software development. Given all these initiatives and processes, it is worth questioning how all of these relate to each other and whether ABN AMRO still has clear sight on all the initiatives and processes running. In an attempt to organize this a little better, we will use our security scope insights from section 5.2 and try to understand how these security initiatives and processes relate in figure 7.3:



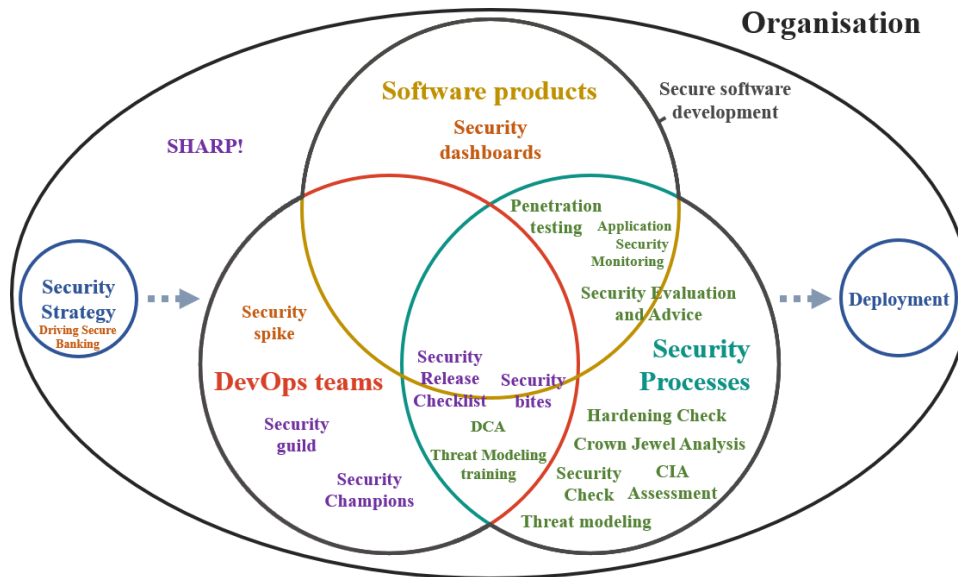


Figure 7.3: The relation of the ABN AMRO initiatives, in purple, and processes, in green, based on our scope insights.

This visualization can be used as a way to structure the large number of security initiatives and processes available at ABN AMRO. It can be valuable for reasoning about the function of the different ABN AMRO security initiatives and processes towards improving the security maturity of DevOps teams. We notice that there are many security processes that cover both security issue finding activities and security environment improving activities (see section 5.2). We also notice that there seems to be a lack of initiatives for training developers about these different CISO security processes. From our discussions with IT security wizards at ABN AMRO during our research period, it became clear that also CISO employees recognize this. Various DevOps teams seem to be unaware of all the CISO processes that exist, why they are important, and how to use them. They often look for approval from CISO, even though CISO’s role is to offer advice. The actual responsibility for security rests with the developers themselves.

As a third observation, in section 7.4.1 we noticed that, even though on first sight it shows some potential, the DevOps Capability Assessment process is not ideal for evaluating the security maturity of DevOps teams. This is because it covers more aspects than just DevSecOps and also has some other potential improvement areas, such as more explicit capability descriptions.

Finally, in section 7.5 we observed that the ABN AMRO Development Analytics Dashboard uses unclear abbreviations, and seems to be incomplete on code quality information of various development teams.

## Chapter 8

# Evaluating the security maturity of mobile application DevOps teams

In the previous chapters (4 to 7), we identified different security maturity evaluation methods, investigated their scopes and we tried to understand the confusing security terminology and the current situation at the ABN AMRO Bank. In this chapter, we will combine our insights to practically carry out a security maturity evaluation with the mobile application DevOps teams at ABN AMRO. For this, we decided to use Weir’s Team Security Assessment [86] in section 8.1, because of its narrow scope that aligns well with our goal to evaluate individual development teams (see section 5.3.3). After that, in section 8.2, we attempt to validate our security maturity results based on available metrics in ABN AMRO’s dashboards, with limited success. Therefore, in section 8.3, we discuss the evaluation interviews that we conducted with two mobile application developers and a chapter lead, to validate our results from applying Weir’s Team Security Assessment, and evaluate the assessment itself. Finally, in section 8.4 we address the limitations of our research, as discussed in 8.1 and 8.3.

### 8.1 Applying Weir’s Team Security Assessment

As mentioned in section 5.3.3, Weir’s Team Security Assessment [86] has most potential for evaluating the security maturity of DevOps teams, opposed to other security maturity evaluation methods (see chapter 4), due to its narrow scope on secure software development. This assessment takes a *qualitative approach* and consists of a questionnaire (appendix A) meant to be completed within 10 minutes (see section 4.3). We decided to practically apply Weir’s Team Security Assessment with development teams of ABN AMRO to see how useful this security maturity evaluation method can be.

### 8.1.1 Considering four maturity evaluation factors

During our literature research, we identified four factors one needs to consider when planning to carry out some maturity evaluation [43]. They can also be considered when planning a security maturity evaluation. For each of the four factors, there are three possible options one can choose from, and each of these options has its own benefits and drawbacks (see tables 8.1 and 8.2). The choices one makes generally depend on aspects like the desired result accuracy, the available time and the available budget for the security maturity evaluation. Therefore, we first considered which combination of these factors, *independence*, *interaction*, *evidence* and *validation* would be applicable to the security maturity evaluation that we conducted. We aimed to establish an efficient evaluation process that allowed us to evaluate multiple DevOps teams within a relatively short time frame.

- *Independence*: Considering the three available options for independence of the assessor from the assessed, our evaluation was a self-assessment. Naturally, this is because Weir’s Team Security Assessment is designed to be a self assessment that participants need to complete. It allowed for a fast evaluation process in which multiple evaluations can be conducted simultaneously, but it required substantial support from the participants and results might have questionable accuracy.
- *Interaction*: When considering the interaction between us as assessor, and the assessed, our evaluation was transactional. We asked the evaluation questions only once and send the questionnaire from Weir’s Team Security Assessment via e-mail to the participants. Again, this allowed for many parallel evaluations, but also substantial participant support was required and results might be questionable.
- *Evidence*: When applying Weir’s Team Security Assessment, the evidence that participants had to provide to support their answers was based on opinion, without requiring actual proofs. Again, this allowed for quick evaluations, but results might be untrustworthy.
- *Validation*: Finally, Weir’s Team Security Assessment does not incorporate a validation process of the maturity results by design, but we decided to try and validate our results with the participants. Particularly, this is because with the options used for the other evaluation factors that we just discussed, chances are that results may be untrustworthy, while we would like to get some assurance in our results.

We will now provide a complete overview in tables 8.1 and 8.2 that shows all of the three possible options for the four different evaluation factors as found in [43]. For each of these options, we highlight both the benefits and drawbacks. The options that align with the security maturity evaluation that we conducted are marked green.

<b>Evaluation factors</b>	<b>Independence</b> <i>(of the assessor from the assessed)</i>		<b>Interaction</b> <i>(between the assessor and the assessed)</i>	
<b>Possible options</b>	Self assessment <i>(The assessed conduct(s) the assessment)</i>		Transactional <i>(Assessment questions are asked once, e.g. via e-mail or spreadsheet)</i>	
	Benefits	Drawbacks	Benefits	Drawbacks
	- Multiple simultaneous evaluations - Fast evaluation process	- Questionable accuracy - Requires substantial support for the assessed	- Many parallel & quick evaluations	- Inaccurate & inconsistent results - Requires substantial support for the assessed
	Internal assessment <i>(Someone inside the organization conducts the assessment)</i>		Interview <i>(Assessor holds interviews with the assessed)</i>	
	Benefits	Drawbacks	Benefits	Drawbacks
	- Consistent results	- Potential bias in results	- In-person, in-depth discussion, clarity - Strong results	- Resource intensive
	External assessment <i>(Someone outside the organization conducts the assessment)</i>		Workshop <i>(Assessor arranges maturity aspects workshops)</i>	
	Benefits	Drawbacks	Benefits	Drawbacks
- External expertise - Comparison data	- Costs - No understanding of the organization; inaccurate evaluation	- In-depth, rigorous results - Collective input and discussion	- Costs of setting up and holding the workshops - Ensuring attendance	

Table 8.1: Compact overview of maturity assessment factors (1) [43]

<b>Evaluation factors</b>	<b>Evidence</b> <i>(used to support the maturity assessment)</i>		<b>Validation</b> <i>(of the maturity results with the assessed)</i>	
	<b>Opinion</b> <i>(Assessment entirely based on opinion without requiring proof for assertions)</i>		<b>None</b> <i>(No validation of results with the assessed)</i>	
<b>Possible options</b>	<b>Benefits</b>	<b>Drawbacks</b>	<b>Benefits</b>	<b>Drawbacks</b>
	- Quick, without evidence collection and verification overheads	- Results may not be trustworthy or validated	- No time or investment required	- No error checking - No feedback from the assessed; could cause friction in the future
	<b>Targeted evidence</b> <i>(Assessor asks for opinion and requests evidence sometimes)</i>		<b>Passive</b> <i>(Assessed are informed of results, but assessor waits for them to raise any concerns)</i>	
	<b>Benefits</b>	<b>Drawbacks</b>	<b>Benefits</b>	<b>Drawbacks</b>
	- Evidence requests can be targeted at units needing more evaluation support	- Difficult balance between accuracy and administrative burden	- Assessed can identify errors or provide feedback, and may be more 'bought-in'	- Relies on the results review of the assessed - No time means no validation
	<b>All evidence</b> <i>(Assessor requires evidence for each assertion made)</i>		<b>Active</b> <i>(Assessor actively asks assessed whether they agree with the evaluation and why)</i>	
<b>Benefits</b>	<b>Drawbacks</b>	<b>Benefits</b>	<b>Drawbacks</b>	
- High assurance degree in results	- Dramatically increased administrative burden for assessed and assessor	- Assurance that results are verified - Error checking exists	- Requires in-depth understanding of assessment criteria - Resource intensive	

Table 8.2: Compact overview of maturity assessment factors (2) [43]

### 8.1.2 Distributing Weir’s Team Security Assessment

After we considered the four maturity evaluation factors, we started distributing Weir’s Team Security Assessment [86] across the Digital Channels Personal Banking grid, which is the collective name for the development teams responsible for the mobile native applications of ABN AMRO. We reached the different teams with help of some security guild members (see section 7.3.5) via Microsoft Teams and e-mail. In the communication, we included a team-based URL that directed to Weir’s Team Security Assessment questionnaire (see appendix A), and we kindly asked to participate in this master thesis research. We chose to distribute the questionnaire to each team individually to ensure distinct reports would be generated for every team. Additionally, we provided the participants with the participant information web page [84] associated with Weir’s Team Security Assessment.

In total, 14 mobile application teams were asked to fill out Weir’s Team Security Assessment questionnaire. The total number of developers in these teams varied, as well as the amount of developers per team that participated in our study. We provide an overview with the number of participants out of the total number of developers for each team in table 8.3. We also include the average amount of time it took each team to complete the questionnaire. The assessment took place from May 25th, 2023 to the 9th of June, 2023.

Team alias	Team name	Number of developers	Number of participants	Time to complete
Crocodile	AA App Service, Interaction & Debit Cards	11	5	9 min
Hammers	AA App Payments 2	9	4	6 min
Iron man	AA App Investment	10	5	3 min
Komodo	AA App Customer interaction and self reliance	9	6	4 min
Lizard	AA App Foundation Authentication & Device Registration	10	6	10 min
Mercury	AA App Messages	10	7	8 min
Phoenix	AA App Personal Finance Management	15	12	4 min
Red Bulls	AA App Foundation Product and Party Onboarding	11	10	29 min
Red Carpet	AA App Foundation Customer Onboarding	11	2	N/a
Scissors	AA App Payments	11	5	15 min
Shield	AA App Architecture Modularisation Refactor	7	9	17 min
Shifu	AA App QA MI Support	7	4	4 min
Spiderman	AA App Overview Consumer Credits Mortgages Pensions	7	5	5 min
Turtle	AA App Foundation Authorisation Security Daily Limits	11	9	21 min
Total amount:		139	89	≈ 10 min

Table 8.3: Per team, the number of participants related to the total number of developers and their average amount of time to complete the assessment

### 8.1.3 Results of our security maturity evaluation of mobile application DevOps teams

Once participants completed the questionnaire of Weir’s Team Security Assessment and the evaluation period concluded, we had thirteen automated Development Team Security-Readiness Reports ready. Each report was of a DevOps team that took part. We noticed the importance of having security enthusiasts with the right authority, e.g. from the security guild, supporting the security maturity evaluation process, to make sure that sufficient developers from the different teams completed the questionnaire. Unfortunately, team Red Carpet did not meet the requirement of having at least 4 participants to keep individual results private. Therefore, no report was generated for this team.

The resulting Development Team Security-Readiness Reports included four informative diagrams, as we discussed in section 4.3. For each of the participating development teams, we extracted these diagrams from the automated reports, and we placed them in a summarizing overview that can be found in appendix C. Most diagrams are intuitive to understand, and do not need further explanation. However, since the first diagram about security importance might be a bit confusing to understand when first looking at it, we will now consider how to read it.

To start with, the vertical Y-axis in this diagram is like a ladder that shows the overall importance of the different aspects of software development for a team, including security. We can rank this importance *top to bottom* with a number from 1 to 7, where a software development aspect with a score of 7 is most important for a team. On the horizontal X-axis, we can find the range of priorities from 1 to 7 that different team members gave to each of the software development aspects, during the security maturity evaluation. Note that here, a *lower* number indicates a *higher* priority. To illustrate, if many team members rated their security priority between 1 and 2 on the X-axis, then this aspect is likely to receive a high score for the overall team security importance on the Y-axis, such as a score of 7.

#### 8.1.3.1 Analyzing our results

Now we can make the following observations based on each of the four result diagrams as presented in appendix C:

- If we consider how important the DevOps teams find security, teams Iron man, Red Bulls, Shield and Scissors score highest with an overall security importance ranking of 7. Especially teams Shield and Scissors seem to have a consensus about how important security is to their team, with a relatively small range of security priority between 1 and 3, and between 1 and 4 respectively. The other two teams have a wider range of responses

about the priority of security, which may suggest a lack of communication between team members about security issues. Another team that shows consensus about priority of security in their team is team Crocodile, which rate this priority between 4 and 5 out of 7. Teams Hammers and Shifu score the lowest on their importance of security, with scores of 1 and 3 respectively. Finally, the remaining teams score relatively high on their overall importance for security too, with scores of 5 and 6.

In conclusion, teams Shield and Scissors are positive outliers, while teams Crocodile, Hammers and Shifu are negative outliers.

- If we consider the awareness and usage of problem finding activities by the mobile teams, we find that all teams show awareness, but limited adoption of the activities, except for two teams. These teams are team Crocodile, who shows only limited awareness, and team Lizard, who shows some adoption of problem finding activities. If we consider the agreement within teams about these results, almost all teams agree with the results, and team Crocodile shows surprising agreement about their limited awareness. On the other hand, there is no agreement within team Lizard, Scissors and Shield about their awareness and adoption of problem finding activities.

In conclusion, there is a chance that teams Lizard, Scissors and Shield are positive outliers, given their disagreement, while team Crocodile is a negative outlier for sure.

- If we consider the awareness and usage of process improving activities by the mobile teams, we find that 8 teams show some adoption, and 5 teams show awareness, but limited adoption. The latter 5 teams are team Iron man, Komodo, Red Bulls, Scissors and Shifu. If we consider the agreement within teams about these results, only 4 teams agree with them. These teams are team Hammers, Komodo, Red Bulls and Shifu.

In conclusion, team Hammers is a positive outlier, while teams Komodo, Red Bulls and Shifu are negative outliers.

- Finally, if we consider the awareness and usage of security education activities by the mobile teams, we find that only 4 teams show some adoption, namely team Iron man, Lizard, Phoenix and Spiderman. All of the other 9 teams show awareness, but limited adoption of security education activities. If we consider the agreement within teams about these results, 5 teams agree with them, namely Red Bulls, Scissors, Shield, Shifu and Spiderman. There was no agreement within the other teams. Based on this, we can only reasonably conclude that team Spiderman is adopting some of the available education activities.

In conclusion, team Spiderman is a positive outlier, while teams Red Bulls, Scissors, Shield and Shifu are negative outliers.



If we now take an overall view of our analysis, we can conclude that most mobile application teams show awareness about the different activities, but adoption is often limited. In particular, teams agree with this finding for the problem finding activities. If we consider the process improvement and security education activities, there is a lot of disagreement within the different teams about their own usage and understanding of these activities. In general, teams Shield and Scissors seem to be most security mature, except for their adoption of security education activities. These teams generally had positive results and showed agreement in the different diagrams. On the other hand, teams Shifu, Crocodile and Red Bulls seem to be the least security mature.

## 8.2 Validating our results with security metrics

As we have seen in section 8.1.1, we used a self assessment for our security maturity evaluation method, with transactional interaction and with evidence that is based on opinion of the participants. These three maturity evaluation factors have as most important drawbacks that the accuracy of the results is be questionable and could be untrustworthy. In section 5.3.3, we already highlighted that combining our qualitative security maturity evaluation results with quantitative security metrics is difficult to achieve, but that we do believe in the possibility to use security metrics about the software products to empirically validate our security maturity evaluation results. Therefore, we will attempt to do this validation now, to get more trust in the security maturity results that we found in section 8.1.3. To provide us with security metrics about the mobile application DevOps teams, we will use the Mobile DevOps Pipelines Dashboard at ABN AMRO, because the Development Analytics Dashboard maintained by CISO does not provide much insight when it comes to these teams (see section 7.5).

We first need to decide which specific metrics we will use to validate our security maturity results. An intuitive choice would be to use the number of open security issues found by scanning tools, such as SonarQube. However, as the introduction section of the report generated by Weir’s Team Security Assessment [86] states, “problems are not equal and some may be unimportant to the organization; one could just be lucky, or unlucky, and get a result very different from what one might normally expect” [98] (see appendix B). Moreover, data about security issues is naturally attributed to applications which might involve more than one DevOps team and complicates measurement of individual DevOps teams. Additionally, the majority of open security issues displayed by the Mobile DevOps Pipelines Dashboard are not attributed to any team, and referred to as “Unowned”. Therefore, validating our maturity results with the open security issues will be difficult.

### 8.2.1 DevSecOps security metrics

Fortunately, there are alternative options of security metrics to consider. This is because the literature offers various lists that contain many different kinds of security metrics. These metrics can be used to measure how well a DevOps team is doing in terms of adopting security activities within their software development lifecycle. We will now discuss different kinds of security metrics that we found in the literature.

When revisiting the Six Pillars of DevSecOps [29], particularly focusing on pillar 6 [30], we find that some of the most important security metrics to monitor in a DevSecOps environment are deployment frequency, vulnerability patch time, percentage code automatically tested, and automated tests per application. Moreover, even though the report specifically dedicated to pillar 6 in the Six Pillars of DevSecOps series has not been published at the time of writing this thesis, we also find an extensive list of common metrics in appendix A of an open draft version of this report [30].

Another valuable resource for DevSecOps security metrics is [94]. This DevSecOps guide offers a large amount of metrics categorized into two types:

1. High-Value metrics: These metrics provide the most critical insight into performance of a DevSecOps platform and should be given priority in implementation. Examples include availability and also deployment frequency.
2. Supporting metrics: These metrics are useful for teams looking to improve their DevSecOps practices. Examples include test coverage and change lead time.

Finally, when considering the scientific literature, we come across the literature study of [75], which identified various DevSecOps security metrics to measure the effective implementation of DevSecOps as well. This research presents a set of metrics based on professional and academic perspectives, such as defect density, the number of adversaries per application, the number of successful deploys to production per month, and the number of issues encountered during red teaming drills.

Due to the large amount of different DevSecOps security metrics described in [29, 30, 75, 94], we will not provide a complete overview of all these metrics here. Please refer to the respective articles to obtain a comprehensive list of all the different security metrics.

## 8.2.2 Available security test coverage data

In section 8.2.1, we have explored many different security metrics that we could potentially use for validating our security maturity results (see section 8.1.3). Two metrics that frequently appeared are security *test coverage* and *deployment frequency*. Taking a closer look at the Mobile DevOps Pipelines Dashboard, we observe that deployment frequency is indeed available. Naturally, this metric is centered around the deployment of the ABN AMRO application, rather than being specific to individual teams. However, since all of the mobile application DevOps teams at ABN AMRO collaborate on this same application, the deployment frequency security metric does not provide much insight when trying to validate our security maturity results at the level of teams. As a result, metrics that are focused on applications do not serve as suitable security metrics for achieving our validation objective.

Another promising security metric we often find in these security metric lists is security *test coverage*. This metric is available in the Mobile DevOps Pipelines Dashboard based on data from SonarQube, and is also present on the level of teams. It represents how much of the software has been covered by automated security tests that try to detect potential problems. We will use this metric to validate our security maturity results of section 8.1.3. Table 8.2.2 provides a snapshot of the available test coverage data for each mobile development team. This data was captured on July 27th, 2023.

<b>Team alias</b>	<b>SonarQube test coverage based on non-commented lines of code (%)</b>
Red Carpet	No data available
Red Bulls	No data available
Shifu	No data available
Crocodile	57,6%
Hammers	63,9%
Iron man	65,3%
Mercury	67,4%
Lizard	68,3%
Komodo	69,7%
Scissors	70,4%
Spiderman	71,7%
Turtle	74,6%
Phoenix	76,0%
Shield	81,9%

Table 8.4: Snapshot of SonarQube test coverage metric for each mobile DevOps team, based on non-commented lines of code, shown in percentages.

If we base ourselves on this one specific metric, we find that team Shield is most security mature in terms of security testing, while team Crocodile is the least security mature. The fact that there is no data available for the teams Red Carpet, Red Bulls, and Shifu, could mean that they are not mature in security testing either, but it might have a different reason too.

### 8.2.3 Comparing test coverage with our maturity results

As described in the previous section 8.2.2, security test coverage measures how much of the software has been covered by automated security tests that look for potential problems. This aligns with security finding activities, one of the four categories that we measured using Weir’s Team Security Assessment. Now, if we look at our results from this category in section 8.1.3.1, we find that teams Lizard, Scissors and Shield were potential positive outliers in security maturity, while teams Crocodile, was a negative outlier. Here, results of teams Shield and Crocodile strongly correspond with our findings in table 8.2.2. This is because based on the code coverage metric, team Shield scores highest on security maturity, while team Crocodile scores the lowest. Team Shifu fits our overall assessment findings, and given that we lack code coverage data for them, this might imply security immaturity.

This comparison provides us with more trust in the results of our security maturity evaluation. It especially validates the aspect about problem finding activities that the evaluation contained. However, it does not validate our results on the other aspects that were measured through Weir’s Team Security Assessment. Unfortunately, the Mobile DevOps Pipelines Dashboard lacked other security metrics that are concentrated specifically on a team-level, and which provide insight into the other aspects that were evaluated using Weir’s Team Security Assessment. As a result, while our maturity results gained more credibility based on our security metric validation, there is room for another layer of validation to add more assurance.

## 8.3 Evaluation interviews with mobile application DevOps teams

After applying Weir’s Team Security Assessment [86] with the mobile application DevOps teams (see section 8.1), we conducted three interviews with members of the mobile application security guild (see section 7.3.5). Specifically, we interviewed an Android developer, an iOS developer, and a chapter lead. The chapter lead is in charge of managing and guiding employees who work in a specific domain, such as Android developers, while also engaging in the daily operational tasks. The interview sessions took between 38 and 50 minutes to complete, and to guide these sessions we created an interview template that can be found in appendix D.

We conducted these evaluation interviews because we had two main objectives. Firstly, we aimed to add an extra layer of validation to our security maturity results by considering the perspective of the security guild members, because our validation through security metrics offered only limited assurance. Secondly, we wanted to evaluate Weir’s Team Security Assessment based on actual participants, to determine whether such an additional security maturity evaluation process would actually be desirable by the DevOps teams at ABN AMRO, given that the bank has many different processes in place already (see section 7.4). Moreover, we wanted to identify if any other issues or suggestions came to light that could help improve the security maturity evaluation method further.

### 8.3.1 Validating our results with evaluation interviews

During our evaluation interviews, we validated whether our results produced using Weir’s Team Security Assessment were consistent with the view that interviewees had on security in the mobile DevOps teams. Overall, the participants indicated that the results were consistent. One of the participants mentioned that they were not really surprised by the amount of disagreement within teams about the usage of different security activities. They explained that many of the activities and controls are actually in place, and suggested that teams contain various members who are unaware of this. For example, the teams do have security pipelines with scanning tools in place, but developers barely use them because they do not know about them. The participant also mentioned that they found the limited adoption of security education activities in the teams not surprising either. They said that to most team members, SHARP [11] (see section 7.3.1) is the only security initiative they consider as source of security education, even though it does not focus on secure software development. In conclusion, this validation based on our evaluation interviews did provide us with more confidence in the generated security maturity results using Weir’s Team Security Assessment.

*“I think [this report] pretty accurately describes what I also see.”*

### 8.3.2 Evaluating Weir’s Team Security Assessment

Next to a validation of our results, we also evaluated Weir’s Team Security Assessment [86] with the participants of our interview sessions. First, we will discuss our findings that relate to the security maturity evaluation process in general. Then, we will discuss our findings related to the questionnaire that is part of Weir’s Team Security Assessment. Finally, we will also consider the opinions about the Development Team Security-Readiness Reports that were automatically generated.

A key initial question that we wanted to find out was whether DevOps teams would actually prefer having another additional security process to evaluate the security maturity of their teams. This question came up especially considering that ABN AMRO already has many different security initiatives and processes in place, of which many DevOps teams seem to be unaware and unsure how to use them (see sections 7.3, 7.4 and 7.7). The answer to this question was yes. According to our interview participants, this process was highly valuable and they felt a strong willingness to adopt it. In particular, two participants highlighted that this feeling came also from the current lack of such a process at ABN AMRO. In terms of rating the usefulness of the security maturity evaluation process, one participant rated Weir’s Team Security Assessment as “good” with 4 points on a 5-point Likert scale, while the other two participants rated it as “very good” with 5 points each.

*“I think this report is definitively useful, and hopefully [it will make] security (...) more important”*

One participant also stated that they would like to repeat the evaluation in a year, and see if any improvements had been made.

*“It would be nice to repeat these maturity measurements in a year for example to see if we got any better”.*

Next, we specifically focus ourselves on the questionnaire used in Weir’s Team Security Assessment. Each of our interview participants indicated that the language used in the questionnaire was very understandable. They liked the absence of overly technical terms, making it very understandable and accessible for all team members.

*“I feel [like] they did a very great job with the language used for the questions.”.*

If we consider the duration that it took to complete the assessment questionnaire, the perspectives of our interviewees differed. On a 5-point Likert scale, one participant rated the duration of the assessment as “acceptable”, but somewhat lengthy. Another participant rated the duration as “good”, and our third participant regarded it as being “very good”. The latter participant indicated that the rewards obtained from investing time in completing the questionnaire are very valuable. All participants shared the consensus that a security maturity evaluation of more than 10 minutes would be too long, and they speculated that teams for which the assessment took, for instance, 29 minutes, were possibly due to individuals leaving the questionnaire open while doing other tasks on their laptop. Additionally, they highlighted a preference for the incorporation of more automation in the assessment process, but they also acknowledged the necessity of the questions in the questionnaire.

Finally, we consider the Development Team Security-Readiness Report that is produced upon completion of Weir’s Team Security Assessment. Again, participants were positive about the report’s appearance, and mentioned that they liked its accessibility. Nevertheless, two participants pointed out that they would have preferred a more concise summary in the resulting report, finding the text overly wordy.

Another participant indicated that they view the report primarily as an informative tool, offering a uniform way to see where each team currently stands. The report was perceived as being a valuable starting point. However, they also indicated that they were missing a clear road map illustrating how to improve specific security activities and address weak points for better results, based on where the team stands now. According to this participant, the aspect of ‘how’ is absent in the resulting report. For example, the report does not offer clarity on how the team could improve their execution of the SAST activity.

Lastly, all three participants agreed that adjusting the evaluation and its resulting reports, such that they contain specific activities for different types of DevOps teams would be beneficial. This is because not all security activities that are best to carry out for one team are necessarily best for another team. They mentioned that currently, the results are not necessarily mobile oriented. For example, activities like reviewing configurations might be better suited for back-end teams, while dealing with application crashes could be more relevant for mobile application teams. By making these adjustments, the report could outline what specific security activities are expected from each type of team.

*“I would say configuration review is something we do not do frequently. I think it is something for back-end.”*

*“It did not feel like it was really mobile oriented.”*

On a 5-point Likert scale that ranges from “very poor” to “very good”, all three participants provided a rating of “good” for the Development Team Security-Readiness Report.

### **8.3.3 Other things that we learned**

During our evaluation interview sessions, we discovered some additional insights that went beyond our initial objectives. Because we consider these findings to be relevant as well, we will discuss these in this section.

To begin with, participants indicated that ABN AMRO lacks a clear definition of DevSecOps within the organization. Mostly, discussions about the topic remain at a high level of generality. As a result, the development

teams base themselves on their past experience at other companies. This means that the responsibility to shape DevSecOps falls on the developers themselves. The participants indicated that having explicit and practical guidelines outlining DevSecOps for ABN AMRO would be valuable, and that it would be appreciated if these could be adjusted specifically to mobile development as well. Interestingly, this observation is similar to what we already observed in section 7.2.

Secondly, an interesting observation that came up during our interviews was that the importance of security is not discussed very often within the department. Developers often assume that their work is inherently secure, and have a primary focus on creating new features. Therefore, the responsibility for security largely falls on the back-end teams. Surprisingly, security is not even considered as a requirement or acceptance criterion in the definition of done when building a software product.

Right now, there are ongoing efforts to change this mindset and to make developers more aware. A simple step like having a checkbox that must be marked off before a software product is launched could make a big difference in making sure security is taken seriously. Our security maturity assessment can also act as a tool to facilitate discussions and encourage to incorporate activities like as secure code reviewing, which is a practice that is currently not being done.

Finally, a participant mentioned that the mobile application teams perceive themselves as being a niche and not really taken into account regarding security. At times, they might feel overlooked. This feeling comes, for example, from the fact that various of the security initiatives at ABN AMRO, such as the Security Champion movement 7.3.4, do not really apply specifically to the mobile developers.

*“I think mobile [development] is really isolated from the rest of the bank. There are some initiatives like security champions, but they are not applicable to us.”*



## 8.4 Limitations

In this section, we discuss the limitations associated with our practical application of Weir’s Team Security Assessment [86] as a method for evaluating the security maturity of DevOps teams. These limitations do not only include limitations associated with our application of the assessment, but also those related to Weir’s Team Security Assessment itself, which we identified because we applied it. The latter limitations are additional to the drawbacks that we already identified during our evaluation interviews as discussed in section 8.3, and could possibly be used as points for improvement as well.

### 8.4.1 Limitations to our security maturity evaluation process

The following limitations should be taken into account when understanding the results of our research:

- Firstly, our evaluation of security maturity focused specifically on thirteen of the mobile application teams that work for ABN AMRO, while the bank has over 700 development teams. It is important to note that the mobile application teams are not representative for the entire bank, which means that the findings of our evaluation cannot be generalized to other DevOps teams in the bank, and how they are doing when it comes to security. Each DevOps team is different and might have their own opinion about a security maturity evaluation process like ours.
- Secondly, it is not possible to verify whether each of the answers provided via the questionnaire of Weir’s Team Security Assessment originated from a separate team member. This is because we sent the questionnaire through e-mail and decided to use transactional interaction with our participants (see section 8.1.1). In theory, participants could have completed the questionnaire multiple times to increase their team’s participation. Additionally, the possibility exists that mistakes happen, like accidentally submitting the questionnaire while some questions were still unanswered.
- Finally, the accuracy of our security maturity results might be questionable, because our security maturity evaluation used the self assessment approach and was based on opinion (see section 8.1.1). As mentioned in section 8.2, even though we gained some assurance, we could not validate all of our results using specific security metrics, which does present a limitation to our results. Nevertheless, some confidence can be placed in our security maturity results, because participants in our evaluation interviews, who are knowledgeable about security in the teams, responded positively to these results.

#### 8.4.2 Limitations of Weir’s Team Security Assessment

Next to these limitations, we also identified some disadvantages associated with Weir’s Team Security Assessment [86] as a result of our practical application:

- Firstly, Weir’s Team Security Assessment does a good job when it comes to adjusting the questionnaire based on the role that a respondent fulfills within their development team, either technical or non-technical. However, if we look at the final results, we are unable to determine how much certain roles, like product owners, impact the maturity of within the development teams. Having such insight could be beneficial, because having developers with low security awareness might be a bigger issue compared to a similar situation with team members that are from the business side.
- Secondly, it is unclear how the Development Team Security-Readiness Reports that are automatically generated by Weir’s Team Security Assessment are composed based on the answers respondents provide via the assessment questionnaire. In particular, we would like to understand how certain questions correspond to specific outcomes visualized in the resulting diagrams. Unfortunately, the website does not provide an explanation for this either. As a result, we consider the process of going from collected responses to the final report as a black-box.
- Thirdly, because Weir’s Team Security Assessment uses a qualitative approach, it does not provide some final, overall maturity score like those found in security maturity models. Having such quantified final score could be useful for easy comparison with different DevOps teams.
- Lastly, as we discussed in section 5.3.3, Weir’s Team Security Assessment is centered around the Agile software development methodology, and does not specifically focus on DevOps. Nevertheless, we believe this is not a big issue, because these two methodologies mostly complement each other effectively.

## Chapter 9

# Future work

Because the field of our research is very broad and extensive, we encountered many different research questions while conducting our research that we decided not to address. This decision was made in order to keep a clear and comprehensible scope for our research. In this chapter, we will discuss these different research questions that could drive future research and provide even more insight into the field of evaluating the security maturity of DevOps teams. We have categorized each of our future research questions in sections 9.1 to 9.6, and we will briefly explain our ideas or motivation behind these different research questions.

### 9.1 Maintaining and improving DevOps team security maturity

The following research questions focus on how to maintain and improve the security maturity of DevOps teams. These questions are closely related to our research field, because after evaluating the security maturity of a DevOps team and understanding their current situation, we naturally start to think about ways to maintain or even increase their security maturity. This is important, because the security maturity of DevOps teams might decline over time. New members often join these teams, while existing members might leave the team or even the company. Therefore, a logical research question to guide further investigation resolving this issue could be:

- *How to prevent a decline in the security maturity of teams due to frequent composition changes in DevOps teams?*

Naturally, one would suggest some kind of consistent security training for new DevOps team members to improve their individual security maturity. However, straightforward follow-up questions to consider would be what kind of training would be effective, and which alternatives there are to improve a team's security maturity. If we look at these questions in a broader

sense, we arrive at the following research question, that could be a key starting point for future research:

- *How to improve the security maturity of DevOps teams?*

During our own research, we identified various suggestions and resources one could consider when addressing the above research question, for example via training. We will now highlight these findings, such that future researchers can eventually use them as a starting point for their own work.

Based on our own research, we suggest to take the different areas we identified in the secure software development scope (see section 5.2) into account when using security training to improve security maturity. Particularly, based on our findings in section 8.3, we advice to focus security trainings on the security activities and software development lifecycle most relevant to the specific type of DevOps team undergoing training. This mirrors how the security maturity evaluation should also incorporate different team types. Security activities that are best for one team are not necessarily best for another. Ideally, you want supportive training and resources to support each DevOps team optimally. For example, security training about container segmentation is not directly relevant for mobile application development teams, but it is for back-end development teams. It is important that different types of development team know the security activities they are expected to follow and understand the organization’s definition of DevSecOps.

In literature, we also encountered various recommendations and security advice resources aimed at improving the security maturity of developers. For example, [101] recommends to focus on the 12 most-used security activities based on their analysis, and include “support [for] the use of components, cloud-based computing and developer-centered security”. Furthermore, [20] takes a psychological perspective and reflects on the challenges involved when trying to improve security behavior. This study found that next to understanding security advice, employees must also be motivated to apply it, which requires changes to attitude and intention. As a result, developers should also understand *why* it is important to become more security mature.

Regarding training materials and security advice resources, one option is to use the free workshop materials [83] and Secure Development Handbook [26] belonging to the Secure Development project of Dr. Charles Weir [85], as discussed in section 4.3. Other seemingly interesting resources that could assist individual developers to improve their secure development activities include the OWASP Application Security Verification Standard (ASVS) [67], the OWASP Proactive Controls List [70], the OWASP top 10 [71], the SANS top 25 [80] and the OWASP DevSecOps Guideline [68]. However, it might also be valuable to consider [15] in connection to these security advice resources.

This work analyzes why these different resources often do not effectively improve one's security maturity and suggests how this can be improved.

Perhaps, being aware of the security maturity level of your team already encourages teams to improve their maturity. Especially in case the security maturity level of your team is low. It could also have unwanted consequences for other development activities. For example, it is possible that someone heavily focusing on boosting their security maturity score might accidentally overlook other important aspects of development, such as responsiveness and scalability of the application. Therefore, exploring this could be interesting in future research too, and could be done using the following research question:

- *Does being aware of one's own security maturity improve the security maturity of DevOps teams, and could this have unwanted consequences as well?*

Finally, in section 7.3.3, we identified ABN AMRO's Security Release Checklist, which contributes to training about various security activities and measures. Therefore, this initiative aims to contribute to improving the security maturity of DevOps teams. However, it would be interesting to verify if the new Security Release Checklist is actually beneficial for these DevOps teams. Considering that there are many different other security initiatives and processes at ABN AMRO, similar questions could be asked for those as well. A research question to consider would be:

- *How effective is ABN AMRO's DevOps Security Release Checklist in improving the security maturity of DevOps teams?*

## 9.2 The security maturity evaluation process

Next to future research about maintaining and improving the security maturity of DevOps teams, we also came up with various research questions that relate to the security maturity evaluation process itself.

Firstly, it is logical to question how often security maturity evaluations should ideally be conducted, and whether it is possible to do them consistently with the different teams. This way, results can be compared over time and improvements become visible. Future researchers could consider questions like:

- *What would be the most effective frequency for periodically repeating DevOps team security maturity evaluations?*
- *How could one repeat the security maturity evaluations of DevOps teams consistently?*

Based on our results in section 8.3, we learned that, even though the Development Team Security-Readiness Reports resulting from Weir’s Team Security Assessment are automatically generated, development teams would like to see an assessment that requires less time to fulfill and incorporates more automation. Therefore, it would also be interesting to investigate whether this is possible by means of the following research question:

- *How to incorporate more automation within a security maturity evaluation method, and what impact does this incorporation have on the method used?*

An additional aspect to investigate could be the four evaluation factors that we considered in section 8.1.1 before carrying out our security maturity evaluations. This is because we applied a specific combination of four evaluation factors during our evaluations, but perhaps using a different combination of these factors for a security maturity evaluation provides different or more reliable results. It would be interesting to investigate their influence on the results, which can be done using the research question:

- *What influence does using a different combination of evaluation factors have on the results of a security maturity evaluation?*

Finally, in section 8.2.1, we found that there are extensive lists of different security metrics that can be used to measure effective implementation of DevSecOps. However, the question which of these metrics are truly useful remains, and particularly, which of them can really contribute to validating the security maturity of individual DevOps teams. For this reason, we formulated the following research question that could be of use in future investigations:

- *Which specific security metrics can be attributed to individual DevOps teams, instead of applications, and can contribute to validating the security maturity of DevOps teams after they have been evaluated?*

### **9.3 Potential influences on the security maturity of DevOps teams**

It could also be interesting to explore other factors that might influence the security maturity of DevOps teams, apart from their security knowledge and usage of security activities. For example, we could check if teams working on older applications differ in security maturity from those working on new applications. Additionally, the different roles within a DevOps team, like the product owner, might influence a team’s security maturity as well. Finding out how these roles matter could improve our understanding of a team’s security maturity and could even help to improve it. To study these potential influences, we consider future research questions like:

- *Does the novelty of the applications that DevOps teams work on influence the results of a security maturity evaluation?*
- *How do the different roles within DevOps teams influence the results of a security maturity evaluation?*

## 9.4 Evaluating and comparing OWASP DSOMM

In section 4.2.2, we discussed that we discovered the secure software development scope of security maturity model OWASP DSOMM [69] too late during our research. Since OWASP DSOMM's scope specifically covers DevSecOps, it could be a promising method to evaluate security maturity of DevOps teams. Therefore, questions future researchers could consider are:

- *To what extent is OWASP DSOMM a useful and effective security maturity model for evaluating the security maturity of DevOps teams?*
- *How do our results from Weir's Team Security Assessment relate to results that one would obtain from applying OWASP DSOMM?*

## 9.5 Internal ABN AMRO analysis

Our research only focused on evaluating the security maturity of ABN AMRO's mobile application DevOps teams. However, there are many other development teams that work for the bank, and it would be interesting to see what the maturity of these teams is in future research. In particular, because some security experts at CISO highlighted that the mobile application DevOps teams are among the best in carrying out security activities within the bank, but our findings suggest that there is still room for improvement. This leads to the following potential research question:

- *How do our security maturity evaluation results relate to the security maturity of other teams within ABN AMRO?*

## 9.6 Other companies

Finally, it could be interesting to explore how companies other than ABN AMRO currently evaluate the security maturity of their development teams. Given the growing number of digital threats and the rising adoption of the DevOps methodology among development teams today, chances are that they face comparable challenges. Therefore, carrying out this future research could offer valuable insights, leading us to the following research question:

- *How do other companies, such as Rabobank, ING, SIG, CIP and Albert Heijn evaluate the security maturity of their DevOps teams?*

# Chapter 10

## Conclusions

In this master thesis, we explored how one could evaluate the security maturity of DevOps teams and we addressed related aspects. This proved to be challenging, due to the very wide-ranging scope and confusing terminology associated with this topic. As a result, we could not cover every detail or address all the questions that arise when studying this area, as we discussed in chapter 9. Nonetheless, we believe that we have produced a comprehensive research that offers various valuable insights.

In this chapter, we will present the four most important conclusions that have emerged from our research. We will do so in sections 10.1, 10.2, 10.3 and 10.4. Finally, in section 10.5, we conclude this master thesis by providing some recommendations to ABN AMRO, based on these four conclusions.

### 10.1 Many methods to evaluate security maturity

The first conclusion that we can derive from our research is that there is an extensive body of literature about methods that can be used for evaluating security maturity (see chapter 4). We categorized them as security frameworks, security maturity models and as a more specific instance of an assessment method named Weir's Team Security Assessment. Some of these security maturity evaluation methods are openly available, while others are proprietary. Also, many of these methods appear to be similar and are based on each other, making it hard to find out what their differences are. If we consider the scope of these different security maturity evaluation methods, we notice that almost all of them focus on evaluating security maturity in general, instead of evaluating the security maturity of individual DevOps teams specifically (see chapter 5). This means that they are not focused on secure software development, but consider security maturity in the scope of an entire organization. As a result, they have only limited practical use in evaluating DevOps team security maturity. However, a promising method



that does focus on secure software development specifically, albeit within the Agile development methodology rather than DevOps, is Weir's Team Security Assessment.

## 10.2 Managing Babylonian speech confusion

As we mentioned in the second sentence of this chapter already, the topic that we are researching covers an extensive scope and contains confusing terminology. Because of this, we noticed that there is a lot of speech confusion in this field, with mixed terminology that is often being used interchangeably. This confusion arises from the large number of different aspects associated with this research area. As part of our research, we aimed to bring clarity to this Babylonian speech confusion in which there is misunderstanding between people, by addressing two challenges that we encountered. We will highlight our insights now, because they can be valuable to anyone that plans to carry out future research in this field.

Firstly, it was challenging to understand the varying scopes of the different security maturity evaluation methods that we encountered during our research. For example, some of these methods focus on secure software development specifically, while others are focused on the security maturity of an entire organization. We tried to resolve this issue by creating the scope diagram visible in figure 5.2, and with our discussion in chapter 5. Additionally, it was challenging to understand how different software development methodologies relate to each other. To resolve this and gain a better understanding, we created figure 3.2, which visualizes the scopes of these methodologies.

Secondly, dealing with the mixed terminology that is often used interchangeably in the field was challenging as well. Explicitly communicating terms like dashboards, frameworks, checklists, practices, activities, and controls in a clear and unambiguous way turned out to be more complicated than we initially assumed. To solve this issue and bring more clarity in the terminology, we established a Glossary at the beginning of this thesis. Additionally, we provided insight into the relation and differences between security practices, activities, controls and measures in chapter 6, by creating figure 6.1.

In conclusion, it is crucial not to underestimate the complexity of the different scopes and confusing terminology related to this research topic. When planning to carry out future research, one should remain thoughtful and very precise about what one means throughout the research process.

### 10.3 Evaluating DevOps team security maturity using Weir’s Team Security Assessment

As we already highlighted in section 10.1, during our research we found that Weir’s Team Security Assessment [86] is a promising method to evaluate the security maturity of individual development teams. Even though it is aimed at Agile development, rather than DevOps, we do not consider this a concern as Agile and DevOps are complementary software development methodologies (see chapter 2). Therefore, we practically applied Weir’s Team Security Assessment [86] with 14 mobile application DevOps teams at ABN AMRO, to evaluate their security maturity. Because this security maturity evaluation method is a self assessment, and we used transactional interaction and evidence based on opinion (see section 8.1.1), leading to a degree of uncertainty of results, we validated our results against the test coverage security metric available through a security dashboard of the bank, related to these teams (see chapter 8). This validation offered us only limited assurance, because it validated just one of the four measured aspects in Weir’s Team Security Assessment. Moreover, no other useful security metrics were available in the dashboard that specifically addressed teams (see section 8.2.3). Therefore, we also conducted evaluation interviews with three members of the mobile application security guild (see section 7.3.5), including two mobile developers and a chapter lead responsible for managing mobile developers with specialized knowledge in a specific domain (see section 8.3). Based on these evaluation interviews, we gained increased certainty in our security maturity results (see section 8.3.1). Additionally, during these interviews, we evaluated Weir’s Team Security Assessment, and found that the security guild members were mostly positive about it, but also identified some areas for improvement within the assessment itself (see section 8.3.2). Finally, we also gained some other interesting insights related to the current feelings of mobile developers about security, which we elaborated upon in section 8.3.3.

In conclusion, Weir’s Team Security Assessment [86] is a practical and efficient method to evaluate the security maturity of individual DevOps teams, that can be completed within 10 minutes and offers automated security maturity results based on self assessment (see section 4.3 and chapter 8). We validated the evaluation’s results to a limited extent using the test coverage security metric and through evaluation interviews with members of the mobile application security guild. The security guild members indicated that they see significant value in using this method on a regular basis, to measure the current security status of the mobile DevOps teams. However, Weir’s Team Security Assessment also has various aspects that can be improved (see sections 8.3.2 and 8.4.2), and carrying out the evaluation requires someone with the right authority supporting the security maturity evaluation process and ensuring that sufficient team members complete the assessment.

## 10.4 Many ABN AMRO security processes and initiatives

Finally, we conclude that the ABN AMRO Bank, and more specifically its CISO department, has a large number of different security-related initiatives and processes in place, and that this complex landscape makes it more difficult to thoroughly understand our research topic (see chapter 7). Our analysis of ABN AMRO's current situation aimed to provide additional context for our thesis, but it also resulted in an organization of all of these different security initiatives and processes that was guided by our security scope insights from section 5.2, shown in figure 7.3. Additionally, we made several observations that may require further consideration by the bank. Since we already summarized these observations in section 7.7, we will not repeat them here.

## 10.5 Recommendations for ABN AMRO

In this section, we will conclude our master thesis by listing some recommendations and takeaways for ABN AMRO, based on our conclusions discussed in the previous sections 10.1 to 10.4.

R1. During our exploration of different security maturity evaluation methods, we did not find some kind of perfect evaluation method to evaluate the security maturity of DevOps teams. However, during the rest of our research we did encounter various elements that one could take into consideration when deciding to use a certain security maturity evaluation method. Perhaps, it would be possible for ABN AMRO to use our insights and to design their own ideal security maturity evaluation method. If this were the case, we would recommend the bank to consider the following aspects that we identified during our research:

- The scope of the evaluation method. The security maturity evaluation method should focus specifically on individual teams and secure software development. Moreover, security metrics should be incorporated in the evaluation result, possibly as a type of automated validation. Additionally, the DevOps software development methodology should be taken into account.
- The evaluation method's language and terminology should be clear and not overly technical. This avoids miscommunication and ensures that all those being assessed understand the questions asked.
- The duration of the assessment, because DevOps teams indicated that an evaluation exceeding 10 minutes would be too long.

- The four maturity evaluation factors as described in tables 8.1 and 8.2, to determine the desired accuracy and costs of the security maturity evaluation.
  - The type of DevOps team, such as mobile or back-end, and the type of application that they develop. One could adjust the evaluation based on the specific security activities that a certain team type is expected to carry out and which fit their software development life-cycle. The security activities that are best for one type of team are not necessarily best for another.
  - The different roles within a DevOps team, because a low security maturity of employees with a non-technical role, such as the product owner, may be less important to the final security maturity result than the security maturity of a back-end developer.
  - A way to display how a DevOps team could improve their security maturity, based on the security maturity evaluation results. For example, one could make use of a final overall security maturity team score or level, as is done with security maturity models. These levels could then have certain descriptions and display how a development team could progress and reach a higher level of security maturity.
- R2. Because we expect that developing some kind of ideal security maturity evaluation method could be quite challenging, we recommend adopting Weirs Team Security Assessment for evaluating the security maturity of DevOps teams until an optimized approach emerges. As we have seen throughout our thesis and in our conclusion section 10.3, this method appears to be the best that we encountered in literature. There is also the unexplored OWASP DSOMM [69] security maturity model that we only discovered later, and we consider as being future research. Weir’s approach allows for quick security maturity evaluations, and produces an automated report of the results. Moreover, the resulting report provides a great starting point for useful discussions within DevOps teams about their current security status. We were also able to validate the results it produces to a certain extent. Lastly, this method focuses on the scope of secure software development, rather than assessing the entire organization, as we discussed in section 5.3.3.
- R3. We recommend the CISO department at ABN AMRO to be very precise and explicit when using the confusing security terminology that we identified in our research. This can prevent many misunderstandings, both among security experts and when communicating with different DevOps teams. One possibility is to refer to our Glossary for the definition of complex or unclear security terms. A fitting quote here is from the Irish playwright George Bernard Shaw: *“The single biggest problem in communication is the illusion that is has taken place”* [51].

R4. If we consider our conclusion about ABN AMRO, and specifically our observation that various DevOps teams seem to be unaware of all the different CISO processes that exist and how to use them, as presented in section 7.7, we would like to recommend reviewing the large number of security processes and initiatives again, including the initiatives aimed at training developers about these processes. This could possibly be beneficial, because the quantity of processes could be overwhelming and unclear to follow for developers. Naturally, this raises the question whether introducing a new process for evaluating the security maturity of DevOps teams would be desirable. However, the results of our evaluation interviews indicate a clear need for such a process (see section 8.3.2). In addition to this, our other observations that we presented in section 7.7 could serve as material for further consideration as well, such as developing a mature definition of what it means to work DevSecOps within ABN AMRO.

# Bibliography

- [1] ABN AMRO. *Blocks List*. URL: <https://clarity.nl.eu.abnamro.com/organisation/list/block> (visited on Mar. 13, 2023). (Internal source).
- [2] ABN AMRO. *DevOps Capability Assessment*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_it-control-tower/SitePages/Capability-Assessment.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_it-control-tower/SitePages/Capability-Assessment.aspx). (Internal source).
- [3] ABN AMRO. *DevOps Capability Assessment*. URL: [https://abnamro.sharepoint.com/:x:/r/sites/intranet-informatie\\_it-control-tower/\\_layouts/15/Doc.aspx?sourcedoc=%7B596D238E-F069-4576-89E6-B3462BCB266A%7D](https://abnamro.sharepoint.com/:x:/r/sites/intranet-informatie_it-control-tower/_layouts/15/Doc.aspx?sourcedoc=%7B596D238E-F069-4576-89E6-B3462BCB266A%7D). (Internal source).
- [4] ABN AMRO. *Driving Secure Banking*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_ciso/SitePages/Strategy/Security-Strategy.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_ciso/SitePages/Strategy/Security-Strategy.aspx). (Internal source).
- [5] ABN AMRO. *Grids Overview*. URL: <https://clarity.nl.eu.abnamro.com/organisation/model/grids> (visited on May 8, 2023). (Internal source).
- [6] ABN AMRO. *Reference material DevOps Capability Assessment*. URL: <https://confluence.int.abnamro.com/pages/viewpage.action?spaceKey=TOWER&title=Reference+material+DevOps+Capability+Assessment>. (Internal source).
- [7] ABN AMRO. *Request DevOps Capability Assessment*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_it-control-tower/SitePages/Request-DevOps-Capability-Assessment.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_it-control-tower/SitePages/Request-DevOps-Capability-Assessment.aspx). (Internal source).
- [8] ABN AMRO. *Security Bites*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_greenlight/SitePages/Security-Bites.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_greenlight/SitePages/Security-Bites.aspx). (Internal source).
- [9] ABN AMRO. *Security Bites*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_greenlight/SitePages/About-the-Security-Champions.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_greenlight/SitePages/About-the-Security-Champions.aspx). (Internal source).

- [10] ABN AMRO. *Security Culture & Transformation*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_greenlight/SitePages/TEAM\(1\).aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_greenlight/SitePages/TEAM(1).aspx). (Internal source).
- [11] ABN AMRO. *SHARP!* URL: <https://aab.powerapp.nl/app/>. (Internal source).
- [12] ABN AMRO. *Shifting left on security*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_it-wayofworking/SitePages/DevOps%20Growth%20Model/Shifting-left-on-security.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_it-wayofworking/SitePages/DevOps%20Growth%20Model/Shifting-left-on-security.aspx) (visited on Apr. 11, 2023). (Internal source).
- [13] ABN AMRO. *Support on the DevOps Capability Assessment*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_it-control-tower/SitePages/Support-on-the-DevOps-Capability-Assessment.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_it-control-tower/SitePages/Support-on-the-DevOps-Capability-Assessment.aspx). (Internal source).
- [14] ABN AMRO. *When you have all controls completed*. URL: [https://abnamro.sharepoint.com/sites/intranet-informatie\\_it-control-tower/SitePages/When-you-have-all-controls-completed.aspx](https://abnamro.sharepoint.com/sites/intranet-informatie_it-control-tower/SitePages/When-you-have-all-controls-completed.aspx). (Internal source).
- [15] Yasemin Acar et al. “Developers Need Support, Too: A Survey of Security Advice for Software Developers”. In: *2017 IEEE Cybersecurity Development (SecDev)*. 2017, pp. 22–26. DOI: [10.1109/SecDev.2017.17](https://doi.org/10.1109/SecDev.2017.17).
- [16] Sap Conversational AI. “How To Approach Security Development Lifecycle (SDL)”. In: (Mar. 2019). URL: <https://medium.com/@SAPCAI/how-to-approach-security-development-lifecycle-sdl-7a3002a534f3>.
- [17] Muhammad Azeem Akbar et al. “Toward successful DevSecOps in software development organizations: A decision-making framework”. In: *Information and Software Technology* 147 (2022), p. 106894. ISSN: 0950-5849. DOI: <https://doi.org/10.1016/j.infsof.2022.106894>. URL: <https://www.sciencedirect.com/science/article/pii/S0950584922000568>.
- [18] Julia Allen and Nader Mehravar. *How to Be a Better Consumer of Security Maturity Models*. Software Engineering Institute - Carnegie Mellon University. Oct. 21, 2014. URL: <https://apps.dtic.mil/sti/pdfs/ADA614299.pdf>.
- [19] Samar Alsaqqa, Samer Sawalha, and Heba Abdel-Nabi. “Agile Software Development: Methodologies and Trends”. In: *International Journal of Interactive Mobile Technologies (iJIM)* 14.11 (July 2020), pp. 246–270. DOI: [10.3991/ijim.v14i11.13269](https://doi.org/10.3991/ijim.v14i11.13269). URL: <https://online-journals.org/index.php/i-jim/article/view/13269>.

- [20] Maria Bada, Angela M. Sasse, and Jason R. C. Nurse. *Cyber Security Awareness Campaigns: Why do they fail to change behaviour?* 2019. DOI: <https://doi.org/10.48550/arXiv.1901.02672>. arXiv: 1901.02672 [cs.CR].
- [21] Kent Beck et al. *Manifesto for Agile Software Development*. Agile Alliance. 2001. URL: <http://www.agilemanifesto.org/>.
- [22] Rich Caralli. *Discerning the Intent of Maturity Models from Characterizations of Security Posture*. White paper of Carnegie Mellon University. Jan. 2012. URL: <https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=58922>.
- [23] Richard Caralli, Mark Knight, and Austin Montgomery. *Maturity Models 101: A Primer for Applying Maturity Models to Smart Grid Security, Resilience, and Interoperability*. White paper of Carnegie Mellon University. Nov. 2012. URL: <https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=58916>.
- [24] Center for Infrastructure Assurance and Security. *The Community Cyber Security Maturity Model*. URL: <https://cias.utsa.edu/research/maturity-model/> (visited on July 25, 2023).
- [25] Center for Internet Security. *CIS Critical Security Controls*. Version 8. May 2021. URL: <https://www.cisecurity.org/controls>.
- [26] Charles Weir. *The Secure Development Handbook. Step by Step to Software Security*. 2018. URL: <https://www.securedevelopment.org/app/download/11319524472/SecureDevelopmentHandbook.pdf>.
- [27] Check Point Research Team. *Check Point Research Reports a 38% Increase in 2022 Global Cyberattacks*. Jan. 5, 2023. URL: <https://blog.checkpoint.com/2023/01/05/38-increase-in-2022-global-cyberattacks/>.
- [28] Cloud Security Alliance. *Six Pillars of DevSecOps Series*. Sept. 9, 2021. URL: <https://cloudsecurityalliance.org/blog/2021/09/09/six-pillars-of-devsecops-series/>.
- [29] Cloud Security Alliance and SAFECode. *The Six Pillars of DevSecOps. Achieving Reflexive Security Through Integration of Security, Development, and Operations*. Aug. 7, 2019. URL: <https://cloudsecurityalliance.org/artifacts/six-pillars-of-devsecops/>.
- [30] Cloud Security Alliance and SAFECode. *The Six Pillars of DevSecOps: Measure, Monitor, Report & Action*. Unpublished draft report. URL: [https://docs.google.com/document/d/1V01z-Fv5UdX0etTX8z6MB8RBgfWEBSG9x\\_w78zIoPSs](https://docs.google.com/document/d/1V01z-Fv5UdX0etTX8z6MB8RBgfWEBSG9x_w78zIoPSs) (visited on July 5, 2023).



- [31] Cloud Security Alliance and SAFECode. *The Six Pillars of DevSecOps: Pragmatic Implementation*. Dec. 14, 2022. URL: <https://cloudsecurityalliance.org/artifacts/six-pillars-devsecops-pragmatic-implementation/> (visited on July 5, 2023).
- [32] CMMC 2.0: A consistent cybersecurity framework. Nov. 22, 2021. URL: <https://www.cmmc-eu.com/cmmc-framework-2-0/>.
- [33] CMMI Institute. *CMMI Levels of Capability and Performance*. URL: <https://cmmiinstitute.com/learning/appraisals/levels> (visited on May 1, 2023).
- [34] Mihai Liviu Despa. “Comparative study on software development methodologies”. In: *Database Systems Journal* 5.3 (2014), pp. 37–56.
- [35] Barbara Ericson. *DevSecOps Best Practices*. Oct. 28, 2022. URL: <https://www.clouddefense.ai/blog/devsecops-best-practices>.
- [36] Brian Fitzgerald and Klaas-Jan Stol. “Continuous software engineering: A roadmap and agenda”. In: *Journal of Systems and Software* 123 (2017), pp. 176–189. ISSN: 0164-1212. DOI: <https://doi.org/10.1016/j.jss.2015.06.063>. URL: <https://www.sciencedirect.com/science/article/pii/S0164121215001430>.
- [37] Gartner. *Definition of Security Metrics*. URL: <https://www.gartner.com/en/information-technology/glossary/security-metrics> (visited on Aug. 15, 2023).
- [38] GitLab. “What is DevOps?” In: *GitLab* (Jan. 2023). URL: <https://about.gitlab.com/topics/devops/>.
- [39] Tom Hall. *Agile vs DevOps. What are the differences and similarities between agile and DevOps?* URL: <https://www.atlassian.com/devops/what-is-devops/agile-vs-devops> (visited on June 25, 2023).
- [40] Tony Hsu. *Hands-On Security in DevOps. Ensure Continuous Security, Deployment, and Delivery with DevSecOps*. July 30, 2018. ISBN: 9781788995504.
- [41] Jez Humble and J. Molesky. “Why enterprises must adopt devops to enable continuous delivery”. In: *Cutter IT Journal* 24.08 (Aug. 2011), pp. 6–12. URL: <https://www.cutter.com/article/why-enterprises-must-adopt-devops-enable-continuous-delivery-416516>.
- [42] Information Security Forum. *The ISF Maturity Model Accelerator Tool*. Version 1.2. URL: <https://www.securityforum.org/solutions-and-insights/the-isf-maturity-model-accelerator-tool/>.
- [43] Information Security Forum. *Time to Grow. Using maturity models to create and protect value*. Sept. 2014.

- [44] Information Systems Audit and Control Association. *CMMI Cybermaturity Platform*. URL: <https://www.isaca.org/enterprise/cmmi-cybermaturity-platform> (visited on July 25, 2023).
- [45] International Organization for Standardization. “Information technology - Security techniques - Information security management systems - Requirements”. In: ISO 27001:2013 (2013). URL: <https://www.iso.org/standard/27001>.
- [46] International Society of Automation. *ISA/IEC 62443 Series of Standards*. Published between 2007 and 2020. URL: <https://www.isa.org/standards-and-publications/isa-standards/isa-iec-62443-series-of-standards> (visited on July 25, 2023).
- [47] ISACA. *COBIT 5 Framework Publications*. URL: <https://www.isaca.org/resources/cobit/cobit-5> (visited on July 5, 2023).
- [48] ISACA. *COBIT Process Assessment Model (PAM): Using COBIT 5*. ISA, 2013. ISBN: 1604202718.
- [49] Mike Jacobs. *Security in DevOps (DevSecOps)*. Nov. 2022. URL: <https://learn.microsoft.com/en-us/devops/operate/security-in-devops> (visited on June 24, 2023).
- [50] Julia Krauwer. *Cyberdreiging zet ondernemers aan tot maatregelen*. Apr. 8, 2022. URL: <https://www.abnamro.nl/nl/zakelijk/insights/sectoren-en-trends/technologie/cyberdreiging-zet-ondernemers-aan-tot-maatregelen.html>.
- [51] Conor Kenny. “The single biggest problem in communication is the illusion that it has taken place”. In: (Nov. 2020). URL: <https://www.irishtimes.com/culture/books/the-single-biggest-problem-in-communication-is-the-illusion-that-it-has-taken-place-1.4404586>.
- [52] L&Co Staff Auditors. *Security Maturity Models: Common Levels of Maturity & How They’re Evaluated*. Feb. 3, 2023. URL: <https://linfordco.com/blog/security-maturity-models/> (visited on May 1, 2023).
- [53] Lanfear. *Secure development best practices on Microsoft Azure*. Feb. 2023. URL: <https://learn.microsoft.com/en-us/azure/security/develop/secure-dev-overview> (visited on June 24, 2023).
- [54] Ngoc T. Le and Doan B. Hoang. *Can maturity models support cyber security?* Dec. 1, 2016. DOI: 10.1109/PCCC.2016.7820663. URL: <https://ieeexplore.ieee.org/document/7820663>.
- [55] Marcus Business Team. *3 Key To Business Success: People, Process & Product*. Oct. 20, 2020. URL: <https://www.marcuslemonis.com/business/3ps-of-business>.

- [56] Mohamed Noordin Yusuff Marican et al. “Cyber Security Maturity Assessment Framework for Technology Startups: A Systematic Literature Review”. In: *IEEE Access* 11 (2023), pp. 5442–5452. DOI: [10.1109/ACCESS.2022.3229766](https://doi.org/10.1109/ACCESS.2022.3229766). URL: <https://ieeexplore.ieee.org/abstract/document/9989381>.
- [57] Joel Martelleur and Amina Hamza. “Security Tools in DevSecOps: A Systematic Literature Review”. MA thesis. Linnaeus University, 2022. URL: <https://lnu.diva-portal.org/smash/get/diva2:1727554/FULLTEXT01.pdf>.
- [58] John Masserini. *Free NIST CSF Maturity Tool*. Jan. 28, 2019. URL: <https://johnmasserini.com/2019/01/28/free-nist-csf-maturity-tool/>.
- [59] Microsoft. *Microsoft Security Development Lifecycle Practices*. URL: <https://www.microsoft.com/en-us/securityengineering/sdl/practices> (visited on Feb. 27, 2023).
- [60] Microsoft. *Microsoft Security DevOps*. URL: <https://www.microsoft.com/en-us/securityengineering/devsecops> (visited on Mar. 22, 2023).
- [61] Microsoft. *What is DevOps? DevOps Explained*. URL: <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-devops/> (visited on June 25, 2023).
- [62] Håvard Myrbakken and Ricardo Colomo-Palacios. “DevSecOps: A Multivocal Literature Review”. In: *Software Process Improvement and Capability Determination*. Ed. by Antonia Mas et al. Cham: Springer International Publishing, 2017, pp. 17–29. ISBN: 978-3-319-67383-7. DOI: [https://doi.org/10.1007/978-3-319-67383-7\\_2](https://doi.org/10.1007/978-3-319-67383-7_2).
- [63] National Institute of Standards and Technology. *Security and privacy controls for information systems and organizations*. SP-800-53 Rev. 5. Sept. 23, 2020. DOI: [10.6028/nist.sp.800-53r5](https://doi.org/10.6028/nist.sp.800-53r5). URL: <https://doi.org/10.6028/nist.sp.800-53r5>.
- [64] National Institute of Standards and Technology. *Security control - Glossary*. URL: [https://csrc.nist.gov/glossary/term/security\\_control](https://csrc.nist.gov/glossary/term/security_control) (visited on July 24, 2023).
- [65] NIST. *NIST Cybersecurity Framework*. Version 1.1. Apr. 2018. URL: <https://www.nist.gov/cyberframework>.
- [66] Office of Cybersecurity, Energy Security, and Emergency Response. *Cybersecurity Capability Maturity Model*. Version 2.1. URL: <https://www.energy.gov/ceser/cybersecurity-capability-maturity-model-c2m2>.

- [67] OWASP Foundation. *OWASP Application Security Verification Standard*. Version 4.0.3. URL: <https://owasp.org/www-project-application-security-verification-standard/>.
- [68] OWASP Foundation. *OWASP DevSecOps Guideline*. Version 0.2. May 16, 2020. URL: <https://github.com/OWASP/DevSecOpsGuideline>.
- [69] OWASP Foundation. *OWASP Devsecops Maturity Model*. URL: <https://owasp.org/www-project-devsecops-maturity-model/> (visited on Mar. 6, 2023).
- [70] OWASP Foundation. *OWASP Proactive Controls*. 2018. URL: <https://owasp.org/www-project-proactive-controls/>.
- [71] OWASP Foundation. *OWASP Top Ten*. 2021. URL: <https://owasp.org/www-project-top-ten/>.
- [72] OWASP Foundation. *SAMM model overview*. Version 2.0. URL: <https://owaspsamm.org/model/> (visited on Mar. 6, 2023).
- [73] Bilge Yigit Ozkan, S.J. Van Lingen, and Marco Spruit. “The Cybersecurity Focus Area Maturity (CYSFAM) Model”. In: *Journal of cybersecurity and privacy* 1.1 (Feb. 13, 2021), pp. 119–139. DOI: 10.3390/jcp1010007. URL: <https://doi.org/10.3390/jcp1010007>.
- [74] PCSI. *Cybersecurity and DevSecOps to a higher level: how major banks and an insurer collaborate*. July 13, 2021. URL: <https://pcsi.nl/news/cybersecurity-and-devsecops-to-a-higher-level-how-major-banks-and-an-insurer-collaborate/>.
- [75] Luís Prates et al. “DevSecOps Metrics”. In: *Information Systems: Research, Development, Applications, Education*. Ed. by Stanisław Wrycza and Jacek Maślankowski. Cham: Springer International Publishing, 2019, pp. 77–90. ISBN: 978-3-030-29608-7. DOI: 10.1007/978-3-030-29608-7\_7.
- [76] PreEmptive Team. “10 DevSecOps Best Practices to Implement Now”. In: *PreEmptive - Professional application protection* (Sept. 8, 2022). URL: <https://www.preemptive.com/10-devsecops-best-practices-to-implement-now/>.
- [77] Rawin Balgobind. *How to: Evidence pages for DevOps Capability assessment*. July 4, 2023. URL: <https://confluence.int.abnamro.com/display/DV/How+to%3A+Evidence+pages+for+DevOps+Capability+assessment>. (Internal source).
- [78] Pilar Rodríguez et al. “Chapter Four - Advances in Using Agile and Lean Processes for Software Development”. In: ed. by Atif M. Memon. Vol. 113. *Advances in Computers*. Elsevier, 2019, pp. 135–224. DOI: <https://doi.org/10.1016/bs.adcom.2018.03.014>. URL: <https://www.sciencedirect.com/science/article/pii/S0065245818300299>.

- [79] W. W. Royce. “Managing the Development of Large Software Systems: Concepts and Techniques”. In: *Proceedings of the 9th International Conference on Software Engineering*. ICSE '87. Monterey, California, USA: IEEE Computer Society Press, 1987, pp. 328–338. ISBN: 0897912160.
- [80] SANS Institute. *CWE/SANS TOP 25 Most Dangerous Software Errors*. URL: <https://www.sans.org/top25-software-errors/> (visited on Mar. 20, 2023).
- [81] Cody Scott. *Assess Your Security Program With Forrester’s Information Security Maturity Model (FISMM). An Extended Functional Assessment*. July 3, 2023. URL: <https://www.forrester.com/report/assess-your-security-program-with-forrester-information-security-maturity-model/RES56671>.
- [82] Secura. *Security Maturity Review*. URL: <https://www.secura.com/uploads/factsheets/Security-Maturity-and-Assessment-Review.pdf> (visited on Mar. 30, 2023).
- [83] Secure Development. *Developer Security Essentials: Workshop Materials*. URL: <https://github.com/SecurityEssentials/Workshops/releases/latest/download/DSEMaterials.zip>.
- [84] Secure Development. *Participant Information*. URL: <https://www.securedevelopment.org/contact/dse-survey-participant-information/>.
- [85] Secure Development. *Secure Software Development*. URL: <https://www.securedevelopment.org/>.
- [86] Secure Development. *Start helping your team with security and privacy*. URL: <https://www.securedevelopment.org/security-assessment/>.
- [87] Mali Senapathi, Jim Buchan, and Hady Osman. “DevOps Capabilities, Practices, and Challenges: Insights from a Case Study”. In: *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*. EASE '18. Christchurch, New Zealand: Association for Computing Machinery, 2018, pp. 57–67. ISBN: 9781450364034. DOI: [10.1145/3210459.3210465](https://doi.org/10.1145/3210459.3210465). URL: <https://doi.org/10.1145/3210459.3210465>.
- [88] Jens Smeds, Kristian Nybom, and Ivan Porres. “DevOps: A Definition and Perceived Adoption Impediments”. In: *Agile Processes in Software Engineering and Extreme Programming*. Ed. by Casper Lassenius, Torgeir Dingsøy, and Maria Paasivaara. Springer International Publishing, 2015, pp. 166–177. ISBN: 978-3-319-18612-2.

- [89] Snyk. “Shift Left Security: Best Practices for Getting Started”. In: *Snyk* (Aug. 2021). URL: <https://snyk.io/learn/shift-left-security/>.
- [90] Joao Souza Neto et al. *A COBIT 5 PAM Update Compliant With ISO/IEC 330xx Family*. Jan. 26, 2018. URL: <https://www.isaca.org/resources/isaca-journal/issues/2018/volume-1/a-cobit-5-pam-update-compliant-with-isoiec-330xx-family>.
- [91] Marco Spruit and G. Slot. “ISFAM 2.0: Revisiting the information security assessment model”. In: *Security Risks: Assessment, Management and Current Challenges* (Jan. 2017), pp. 87–108.
- [92] Michael Swanagan. *The 3 Types Of Security Controls (Expert Explains)*. Dec. 7, 2020. URL: <https://purplesec.us/security-controls/> (visited on July 24, 2023).
- [93] Synopsys. *Building Security Maturity Model (BSIMM)*. URL: <https://www.synopsys.com/software-integrity/software-security-services/bsimm-maturity-model.html> (visited on Mar. 6, 2023).
- [94] Tech at GSA. *DevSecOps Guide. Standard DevSecOps Platform Framework*. URL: [https://tech.gsa.gov/guides/dev\\_sec\\_ops\\_guide/](https://tech.gsa.gov/guides/dev_sec_ops_guide/) (visited on July 5, 2023).
- [95] TNO. *Cybersecurity and DevSecOps to a higher level*. July 22, 2021. URL: <https://www.tno.nl/en/newsroom/insights/2021/07/cybersecurity-devsecops-higher-level-how/>.
- [96] Julien Vehent. *Securing DevOps. Security in the Cloud*. Manning, Aug. 24, 2018. ISBN: 9781617294136.
- [97] Geoff Wagner. *Agile and DevOps: Complementary Approaches to Software Development*. Blog titled A DevOps Blog. Mar. 9, 2023. URL: <https://www.valewood.org/agile-and-devops/>.
- [98] Charles Weir, Ingolf Becker, and Lynne Blair. “A Passion for Security: Intervening to Help Software Developers”. In: *International Conference on Software Engineering* (May 25, 2021). DOI: [10.1109/icse-seip52600.2021.00011](https://doi.org/10.1109/icse-seip52600.2021.00011).
- [99] Charles Weir, Ingolf Becker, and Lynne Blair. “Incorporating software security: using developer workshops to engage product managers”. In: *Empirical Software Engineering* 28.2 (Dec. 24, 2022). DOI: [10.1007/s10664-022-10252-0](https://doi.org/10.1007/s10664-022-10252-0). URL: <https://link.springer.com/content/pdf/10.1007/s10664-022-10252-0.pdf>.

- [100] Charles Weir, Awais Rashid, and James Noble. “Challenging software developers: dialectic as a foundation for security assurance techniques”. In: *Journal of Cybersecurity* 6.1 (Sept. 2020), tyaa007. ISSN: 2057-2085. DOI: [10.1093/cybsec/tyaa007](https://doi.org/10.1093/cybsec/tyaa007). eprint: <https://academic.oup.com/cybersecurity/article-pdf/6/1/tyaa007/33746013/tyaa007.pdf>. URL: <https://doi.org/10.1093/cybsec/tyaa007>.
- [101] Charles Weir et al. “Infiltrating Security into Development: Exploring the World’s Largest Software Security Study”. In: *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. ESEC/FSE 2021*. Athens, Greece: Association for Computing Machinery, 2021, pp. 1326–1336. ISBN: 9781450385626. DOI: [10.1145/3468264.3473926](https://doi.org/10.1145/3468264.3473926). URL: <https://doi.org/10.1145/3468264.3473926>.
- [102] Gregory B. White. “The Community Cyber Security Maturity Model”. In: *2007 40th Annual Hawaii International Conference on System Sciences (HICSS’07)*. 2007, pp. 99–99. DOI: [10.1109/HICSS.2007.522](https://doi.org/10.1109/HICSS.2007.522).
- [103] Wikipedia contributors. *Capability Maturity Model Integration*. Apr. 15, 2023. URL: [https://en.wikipedia.org/wiki/Capability\\_Maturity\\_Model\\_Integration](https://en.wikipedia.org/wiki/Capability_Maturity_Model_Integration) (visited on Apr. 30, 2023).
- [104] Wikipedia contributors. “Waterfall model”. In: *Wikipedia* (May 18, 2023). URL: [https://en.wikipedia.org/wiki/Waterfall\\_model](https://en.wikipedia.org/wiki/Waterfall_model) (visited on June 25, 2023).
- [105] Laurie Williams. “Agile Software Development Methodologies and Practices”. In: *Advances in Computers*. Ed. by Marvin V. Zelkowitz. Vol. 80. Advances in Computers. Elsevier, 2010, pp. 1–44. DOI: [https://doi.org/10.1016/S0065-2458\(10\)80001-4](https://doi.org/10.1016/S0065-2458(10)80001-4). URL: <https://www.sciencedirect.com/science/article/pii/S0065245810800014>.
- [106] John Willis. *What Devops Means to Me*. July 16, 2010. URL: <https://www.chef.io/blog/what-devops-means-to-me>.
- [107] Marta Wilson. “Perfecting People, Process, And Product: Enhancing Organizational Mastery”. In: (Oct. 8, 2020). URL: <https://www.forbes.com/sites/forbesbooksauthors/2020/10/08/perfecting-people-process-and-product-enhancing-organizational-mastery/>.

## Appendix A

# Weir's Team Security Assessment

On the next pages, we included the questionnaire from Weir's Team Security Assessment [86] as discussed in section 4.3, which the participants of our research filled out. The specific list of questions that participants got to see depends on the role the participant had in the development team. In case a participant indicated to have a technical role, they received the questions visible on page 88 and 89. If the participant indicated they did not have a technical role, and they were a product owner for example, they received the questions listed on page 90. The questions in the questionnaire varied from very practical, factual and technical questions, to more process related questions and open questions that asked for the opinion of the participant.





## Welcome to the Developer Security Essentials Team Survey

Please would you kindly complete this short survey questionnaire, to help identify training improvements for you and your team.

The link has been sent to yourself and other colleagues in your team, [Team Name]. A report will be sent to the person who set this up, but only provided that sufficient responses have been received to ensure that individual responses are anonymous. The report is designed to support training and similar security-improvement activities; it can't be used to 'score' teams or individuals.

**To start, please would you confirm (by clicking the red arrow below) that:**

- You have seen the [participant information page](#) and understand what is expected of you within this study.
- Your participation is voluntary.
- You consent for your responses to be used anonymously in a report sent to other members of your team.
- You consent for your responses to be discussed and used in future reports, academic articles, publications or presentations by the research team at Lancaster University and UCL, and you understand that your personal information will not be included and that you and your organisation will not be identifiable.
- You consent to Lancaster University keeping the anonymised data for a period of 10 years after the study has finished.
- You consent to taking part in the current study.



Please rank the following aspects in order of importance for your team when developing software (or a recent project you have worked on). The top item is the most important item.

Available

Responsive

Features

Scalable

Privacy respecting

Easy to use

Secure

---

Do you have a technical role in the development team [Team Name]?

Yes (e.g. Programmer, QA, Team Lead, Architect)

No (e.g. Product Owner, Senior Manager, etc.)



Considering the last few projects you have worked on as part of your team [Team Name], how frequently has the following been the case for the team as a whole?

	In every project	In some projects	irregularly	never	Don't know
You have one member in your development team who specialises in security and privacy.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You get regular, automated reports from an external service on the security of your products.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your formal code reviews include checks for security issues.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You use pen-testing tools internally in your organisation.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The team understand what are the main security risks for your project.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your team is involved in simulations of cyber incidents on your projects to ensure everyone knows what to do.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Nothing gets into the mainline until it's been checked by someone else who knows about security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You use fuzzing as part of your testing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Near the start of a new project, you have a workshop to discuss what are the main security and privacy priorities.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The team understands what are the main privacy risks for your project.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

On a more practical level, how frequently do the following occur in the team?

	Every commit	Every minor release	Every major release	Irregularly	Never	Don't know
You are sure all your components and frameworks are safe against cyberattack.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your team simulate cyber attacks on your software.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You help check other people's code for security issues.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You ensure that your code analysis tools show no security related errors.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You get an external company to check your code for security issues.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your code gets pen tested.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Thinking about the most recent project you have worked on, please consider the following statements, and indicate to what extent you agree to them.

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
None of your components do anything that might worry a user about their privacy.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You all know who to call if there's a cybersecurity incident.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You use automated testing approaches for security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Please give us an introduction to your teams processes regarding software dependencies:

---

How do you manage your software stack dependencies?

When do you decide to update components?

We would like to explore how you, your team and your organisation interact when developing software. Please consider the following statements, and indicate to what extent you agree to them.

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
In your view, your organisation makes an informed decision on which security and privacy improvements to prioritise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Product Owners regularly discuss security improvements and issues with technical staff.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your publicity team know what to do if there's a bad news story about your security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When company-internal security requirements hinder you from working effectively, you can get it resolved.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You know how to handle your customer(s) when there is a security issue.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Thinking more broadly, to what extent do you think the following statements apply to your organisation?

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
Every project has a standard process for security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You have useful standard lists of security threats to consider in each new project.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You are aware of your organisation's contingency plan if there's a security problem that affects your users.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Aspects of Security and Privacy are selling points for your products.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your company supports a security enthusiast in each team with extra security training and support.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



We would like to explore how you, your team and your organisation interact when developing software. Please consider the following statements, and indicate to what extent you agree to them.

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
In your view, your organisation makes an informed decision on which security and privacy improvements to prioritise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Product Owners regularly discuss security improvements and issues with technical staff.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your publicity team know what to do if there's a bad news story about your security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When company-internal security requirements hinder you from working effectively, you can get it resolved.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You know how to handle your customer(s) when there is a security issue.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Thinking more broadly, to what extent do you think the following statements apply to your organisation?

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
Every project has a standard process for security.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You have useful standard lists of security threats to consider in each new project.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
You are aware of your organisation's contingency plan if there's a security problem that affects your users.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Aspects of Security and Privacy are selling points for your products.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Your company supports a security enthusiast in each team with extra security training and support.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



## Appendix B

# Weir's Team Security Assessment sample report

In this appendix, we have included a sample report that is generated automatically based on the results obtained from a team completing the questionnaire of Weir's Team Security Assessment [86], as discussed in section 4.3 and presented in appendix A. This report provides an insight into the typical format and the content of reports generated by Weir's Team Security Assessment, without it specifically representing a certain team from ABN AMRO. For each of the teams we assessed in our research with a minimum of 4 participants responding, we gathered such a report. Especially the generated figures offer valuable insight into a team's current security maturity, and they can facilitate meaningful discussions with the product owner and managing leads in a development team.



# Development Team Security-Readiness Report Prepared for



**Authors:** Developer Security Essentials Team  
**Contact:** [essentials@secureddevelopment.org](mailto:essentials@secureddevelopment.org)  
**Date:** 13 May 2023

## Summary

Our recent survey of your development team, based on 5 surveys received back, provides a self-reported description of the team's skill and adoption levels of the dozen most cost-effective security practices. The responses showed a moderate consistency, suggesting a range of understanding of the security aspects of the team's work.

Analysis of the responses showed the responders mostly agreed on the importance of security and privacy, with a range of responses rating the priority of security between 'fifth' and 'first' of seven kinds of feature.

In terms of the team's security practice, the survey found generally some adoption of technical approaches to find security issues. The survey respondents showed typically general adoption of ways to improve the processes, and also some adoption of techniques for security learning and improvement.

We recommend as your next step to do the Developer Security Essentials workshop, to help the team identify and communicate to each other the motivations and benefits of security and privacy practice.

## Introduction

How do we assess the security needs of a software development team? We could measure outcomes: how many security and privacy problems occur in the finished software. But there are many problems with that: it takes a long time to get a meaningful answer; problems are not equal and some may be unimportant to the organisation; one could just be lucky, or unlucky, and get a result very different from what one might normally expect.

Instead, a much better approach is for us to measure the security-related activities and understanding that the team may have. Based on 5 years of research, including 2 years of interviewing developer security experts, we have identified 'practices' that lead to better security, most of which are possible without specialist knowledge and all of which are suitable for agile development in any size of company [2]. All are explained in the next section. Some of these practices are well-known security Assurance Techniques, such as using a tool to scan source code for some kinds of problem (Automatic Static Analysis); others reflect recent research and observation of successful teams, such as business-focused discussion of security issues with product management and other teams (Product Negotiation). The approach to measurement has been defined over several years of research, and accepted by peer review [1].

The questions completed by the team assessed their understanding and engagement with these techniques. Each team member received an online survey request; the next section analyses the results of that survey.

## Results

A total of 5 questionnaires were completed between 28 September and 29 October 2021, and team members typically took 10 minutes to complete them.

### Importance of Security and Privacy

First, to give context, the team were asked to rate the importance of different requirements including security and privacy in their development. There is, of course, no 'right' answer, but a wide range of answers may suggest a lack of communication between team members about security and privacy issues; it may also be helpful to compare the team's understanding with the organisation's overall strategy.

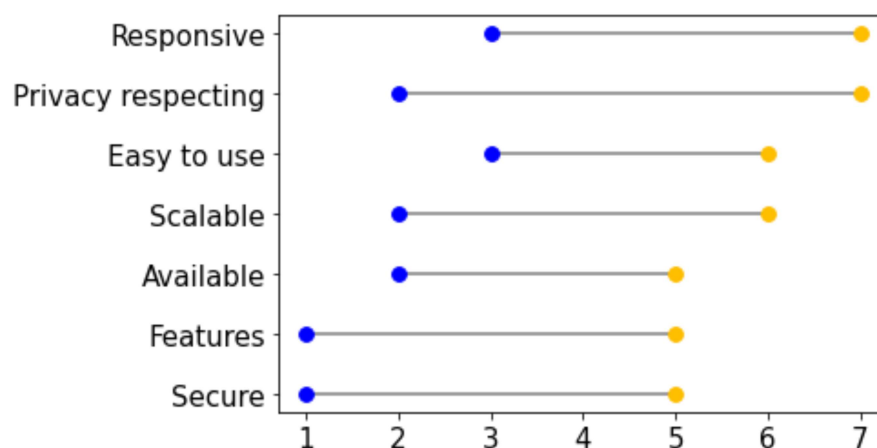


Figure 1: Priorities as identified by team members

Figure 1 shows the importance reported by the team for different requirements. The lines show the range of responses found by the team between the amber (minimum) and blue (maximum) values. As shown, overall, security comes in at about number 7, and privacy at number 2, and team members answers on the ranking of security ranged from 'first' to 'fifth' of the seven.



## Assurance Techniques Used by Development Teams

The majority of questions in the survey address the teams understanding and use of various techniques. For better understanding, we divide the techniques into three groups:

- Problem Finding:** Techniques to find specific security and privacy problems ('vulnerabilities') in created software;
- Process Improvement:** Techniques to create an environment to better support the creation of secure code or reduce the impact of security issues; and
- Education:** Approaches to teach participants and stakeholders about the previous techniques.

Not every technique is appropriate for any given project; often practical, resource or other considerations are good reasons not to use them. So, in the survey, we are interested not only in whether the technique is used, but also whether it has been considered and rejected, and indeed whether the survey participants are aware of the technique and what it implies.

For each technique, we analysed each survey response from the team to give a 'score'. We interpret the scores in the following range:

- Unaware** The technique was apparently not known by the participants
- Aware** Knowledge and understanding showed of the technique. Perhaps there are existing plans to incorporate the technique into development, or the technique has been considered and rejected.
- Well-used** The technique is used (or considered for use) in every new project.

Each score was calculated from the results of several questions in the survey. In representing the scores, we take into account that are usually a range of assessments from the participants. We therefore use a 'violin-style' diagram, to indicate this range.

The next sections explore the survey results for each group of techniques in turn.

### Problem Finding Techniques

The first set of techniques we consider are the ones most familiar to 'security specialists': ways to find 'vulnerabilities' in created software. Specifically, these are:

- Automated Penetration Testing** Using an automated tool to look for common, easily exploited, vulnerabilities in a website or web service.
- Automated Static Analysis** Using automated tools to look for common vulnerabilities in source or binary code.
- Configuration Review** Choosing secure components and frameworks, and keeping them up to date
- Code Review** Scheduled meetings or pair programming to analyse code for security defects

## Penetration Testing

Having a Security Specialist look for vulnerabilities accessible via the web.

Of these, probably the 'easiest' and most cost-effective in many cases is 'Configuration Review', especially if an automated tool can be used to report insecurities.

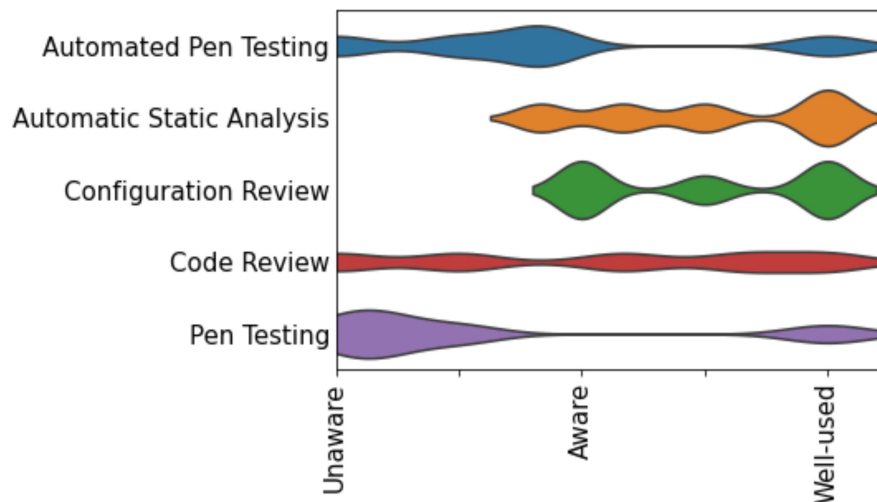


Figure 2: Assessment of the Problem Finding Techniques

Figure 2 shows the team's current assessments for these five techniques. As shown, the team showed generally did not agree about the techniques, and typically reported some adoption of the approaches to find security issues.

## Process Improvement Techniques

The next set of techniques we evaluated are those that improve the development process so as to avoid security problems in the first place:

- Threat Assessment** Design-level analysis of possible attackers, motives, and vulnerability locations (a.k.a. Threat Modelling).
- Product Negotiation** Empowering product management to make security decisions.
- Contingency Plan** The advance creation of a plan to handle security incidents.
- Security Champion** Having a development team member, not usually a security expert, with a particular interest in security. They act as the go-to person for security issues within the development team.
- Standardisation** The creation of standard security configurations, ways of working, or 'Secure Development Lifecycles', plus auditing processes to validate these.

Of these, probably the most powerful and most cost-effective is 'Threat Assessment', since it allows a team to focus on the security issues that are likely to have most impact.

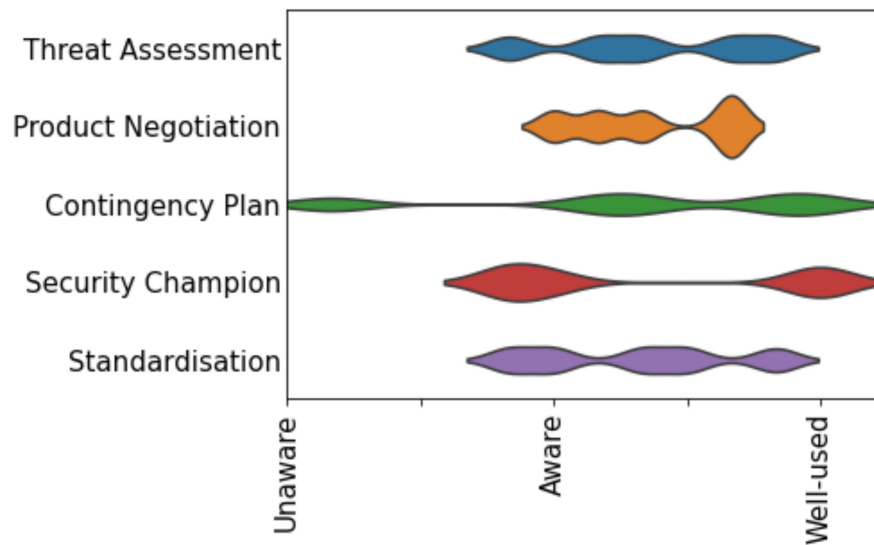


Figure 3: Assessment of Process Improvement Techniques

Figure 3 shows the team's current assessments for these five techniques. As shown, the team showed mostly agreed about the techniques, and typically reported general adoption of the approaches to find security issues.

### Education Techniques

Finally there are training activities to improve the team's knowledge and ability at specific security topics:

- On-the-job Training**      Mentoring or informal workshops, used regularly with the development team
- Further Workshops**      Using focussed workshops for teams to find their own ways to improve.

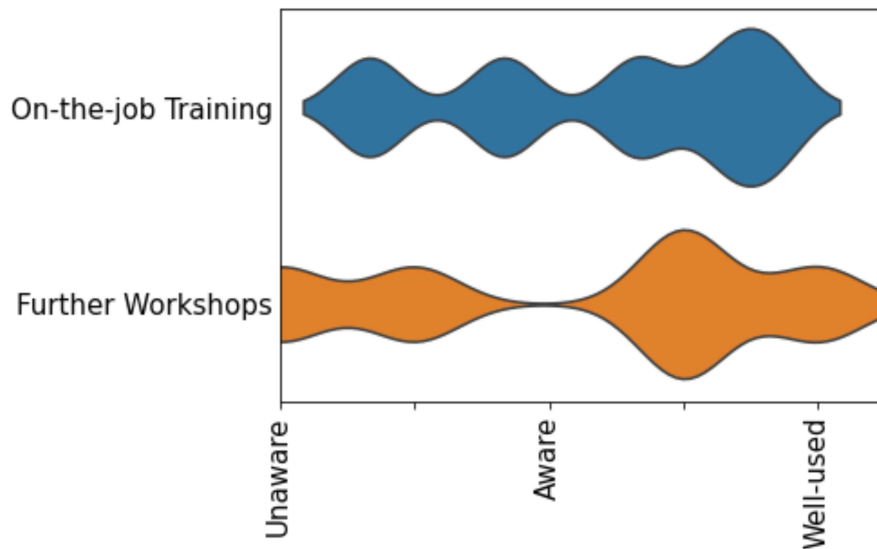


Figure 4: Assessment of Process Improvement Techniques

Figure 4 shows the assessments for these two techniques. As shown, the team generally did not agree about the techniques, and reported some adoption of the approaches to find security issues.

## Next Steps

Naturally, we urge you to complete the Developer Security Essentials workshops with your teams as soon as possible. Follow the link here for more information:

<https://www.securedevelopment.org/workshops/>

For other support adopting the assurance techniques, we have put together a set of resources at this page: <https://www.securedevelopment.org/resources/further-reading/>.

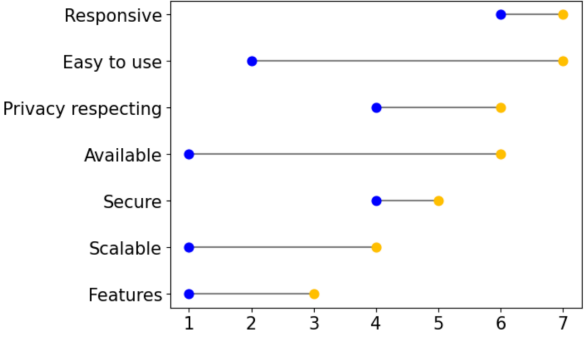
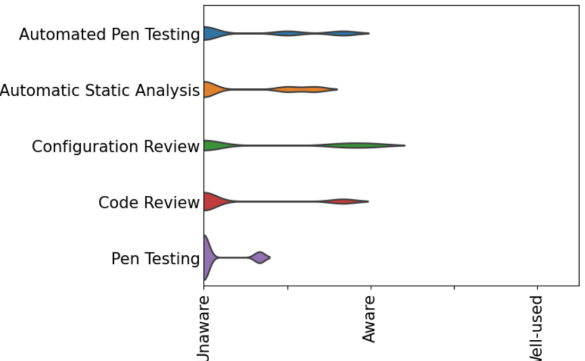
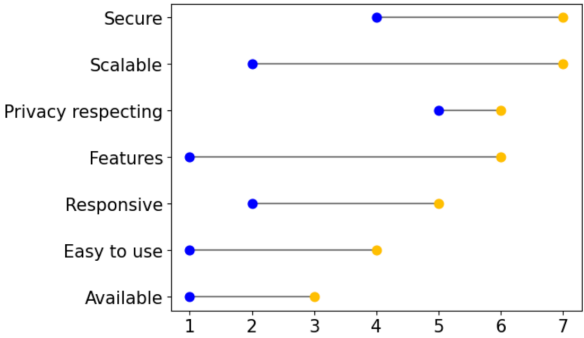
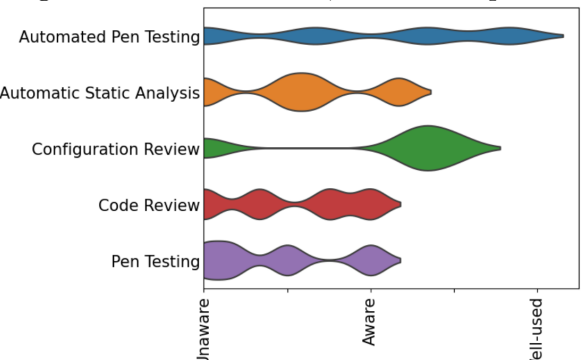
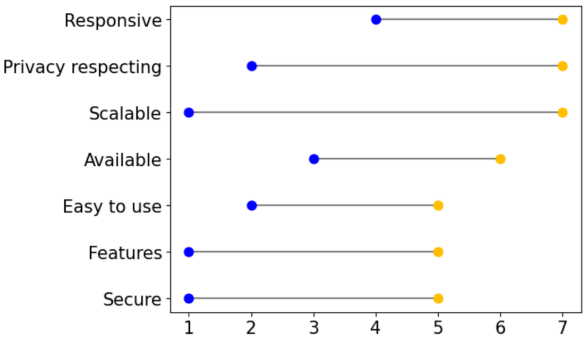
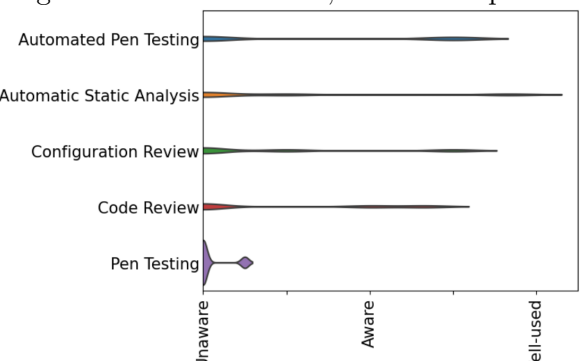
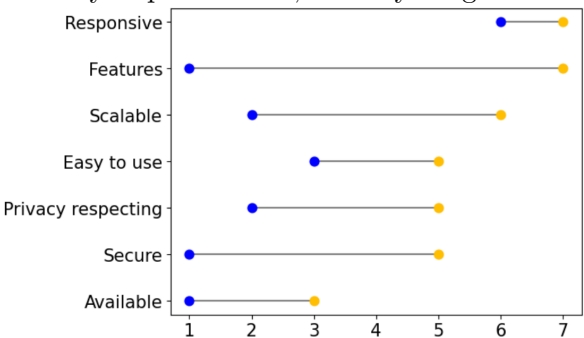
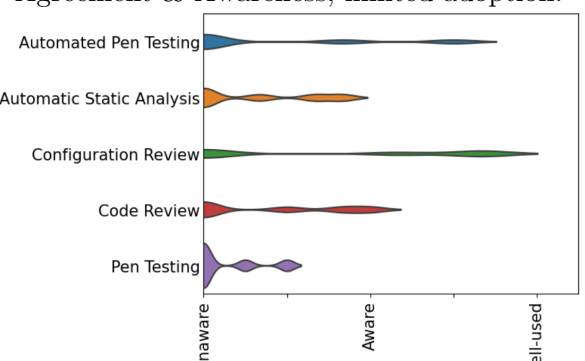
## References

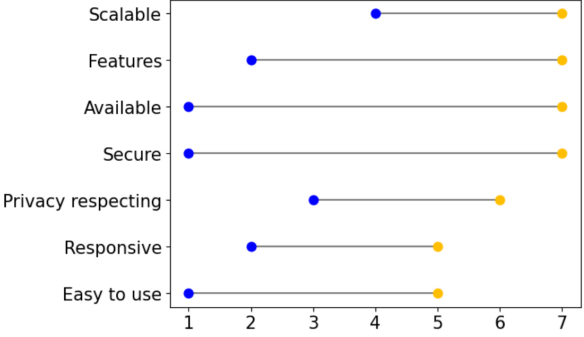
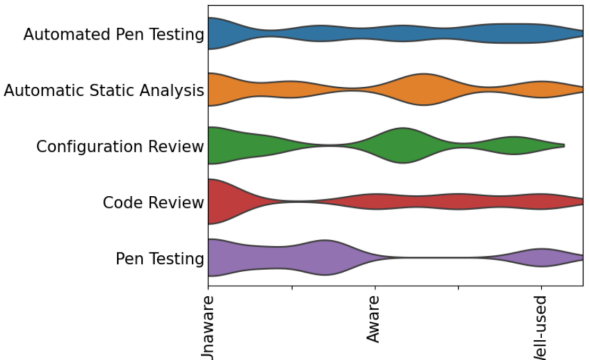
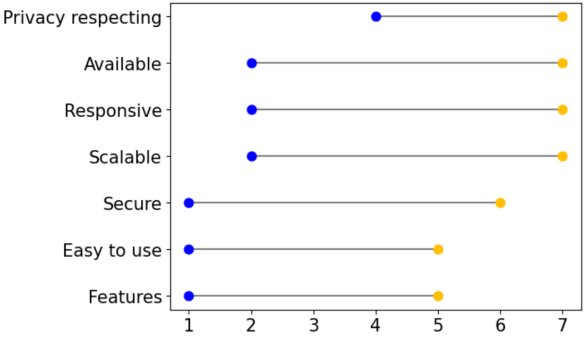
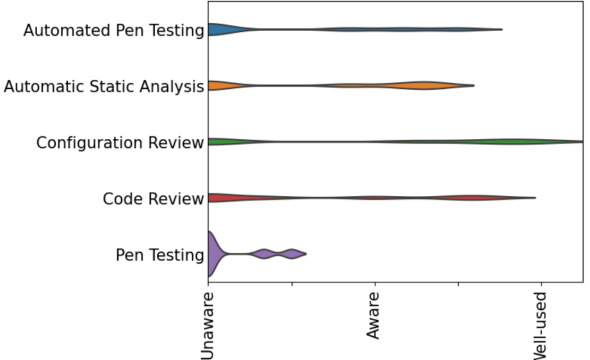
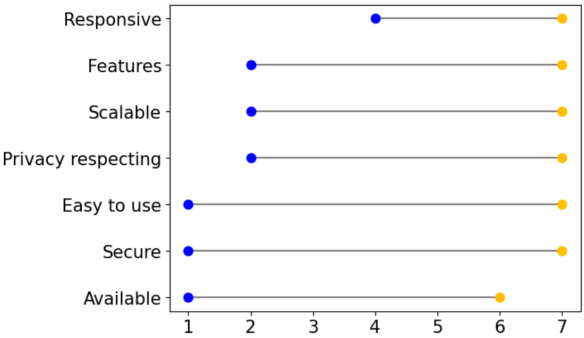
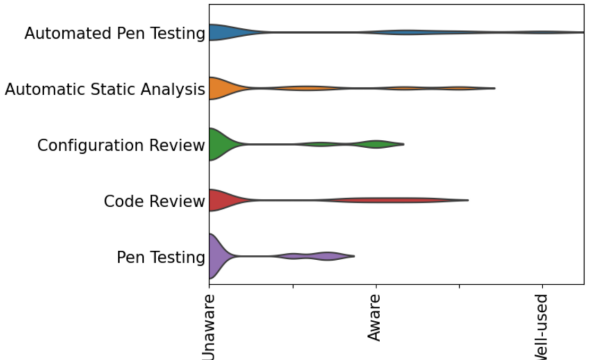
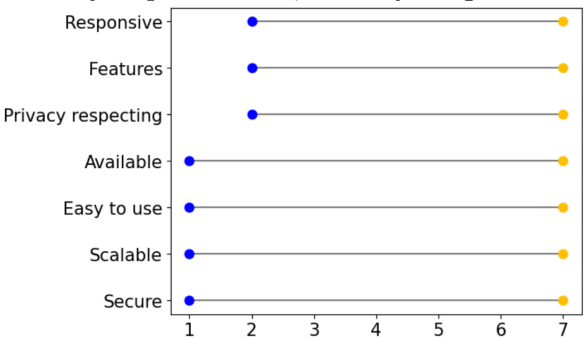
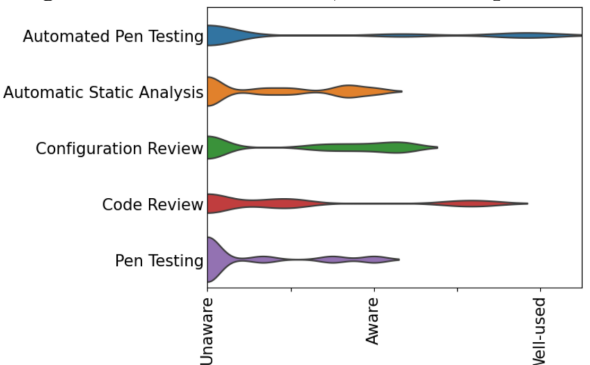
- [1] Weir, C., Becker, I., and Blair, L. A Passion for Security: Intervening to Help Software Developers. *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, IEEE (2021).
- [2] Weir, C., Noble, J., and Rashid, A. Challenging Software Developers: Dialectic as a Foundation for Security Assurance Techniques. *Journal of Cybersecurity*, (2020), 30.

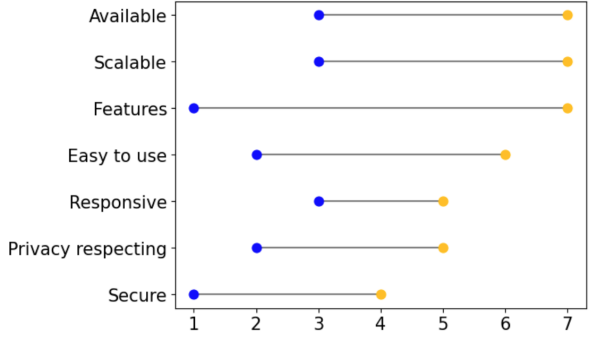
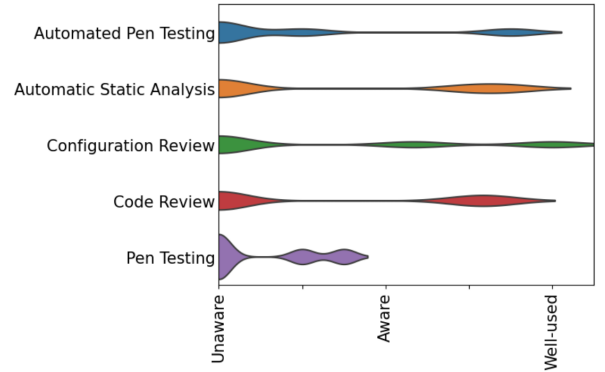
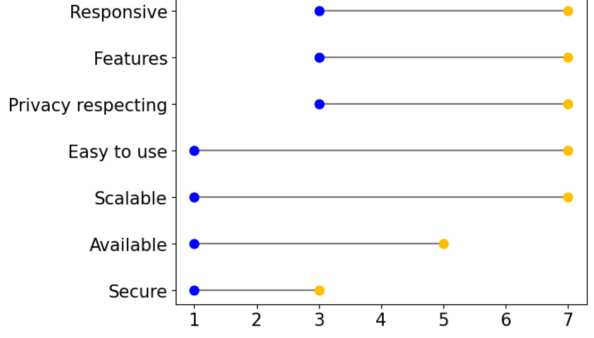

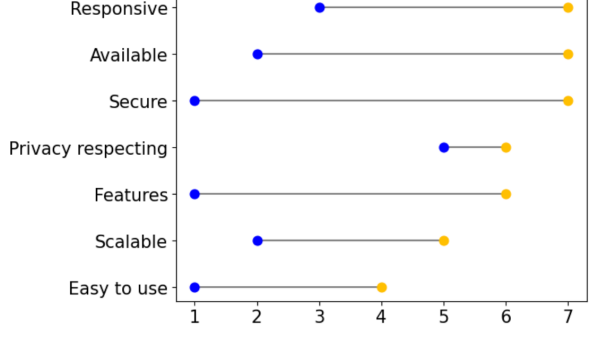
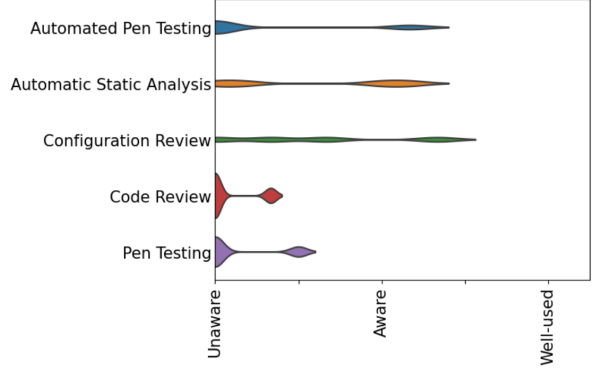
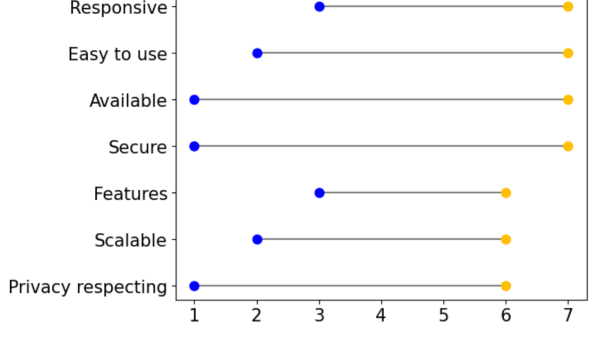
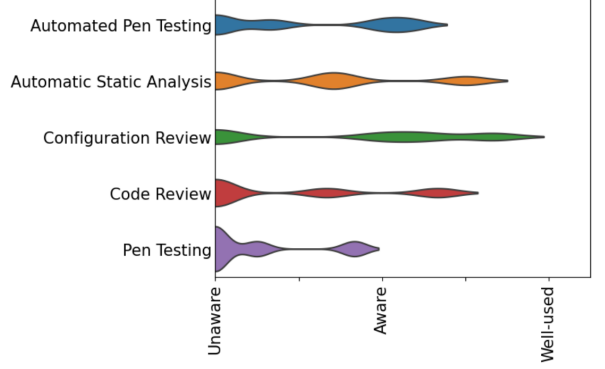
## Appendix C

# Weir's Team Security Assessment results of mobile application DevOps teams

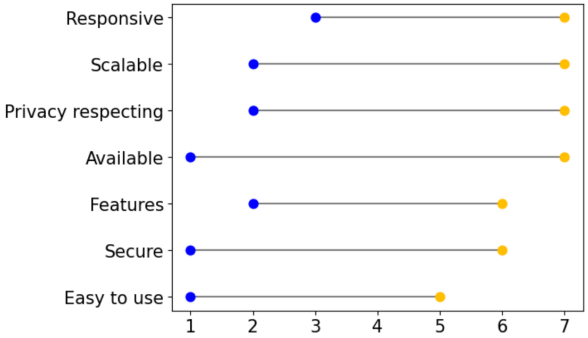
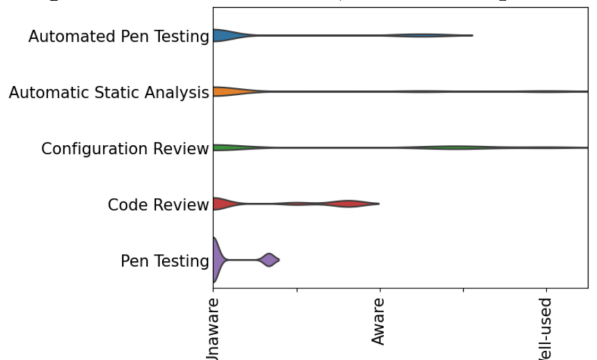
This appendix contains the results of the security maturity evaluations that we have carried out using Weir's Team Security Assessment with 14 mobile application DevOps teams at ABN AMRO (see chapter 8). The results are extracted from the Development Team Security-Readiness Reports that were generated for each team, and displayed in two tables. The first table displays the results about security importance and problem finding techniques for each mobile application DevOps team. The second table displays the results about process improvement techniques and education techniques for each mobile application DevOps team.

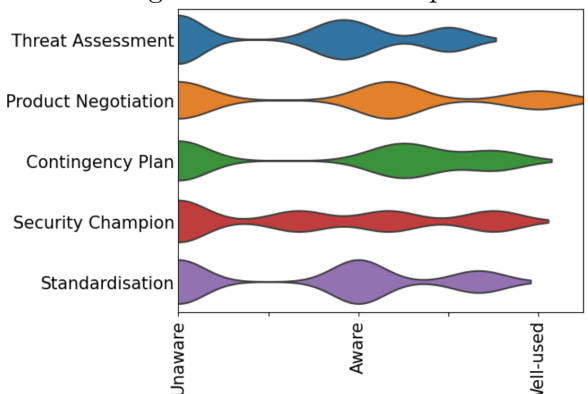
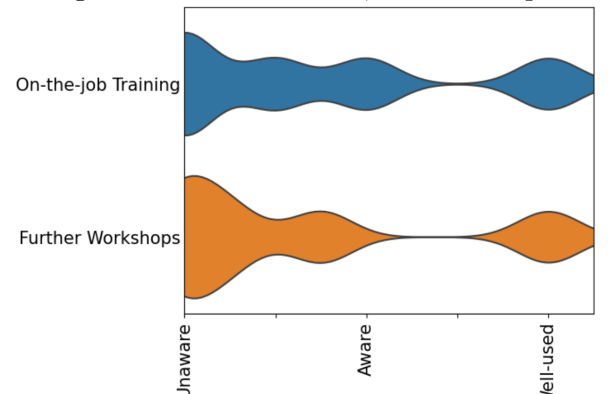
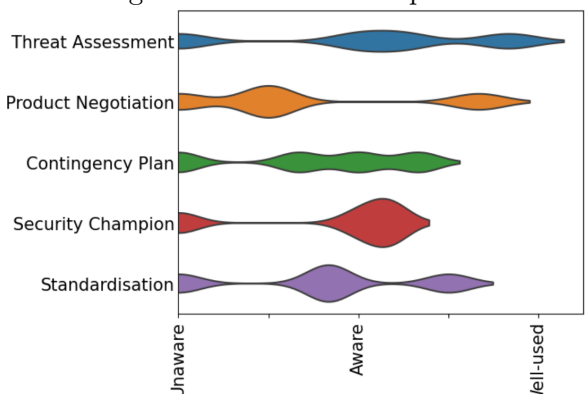
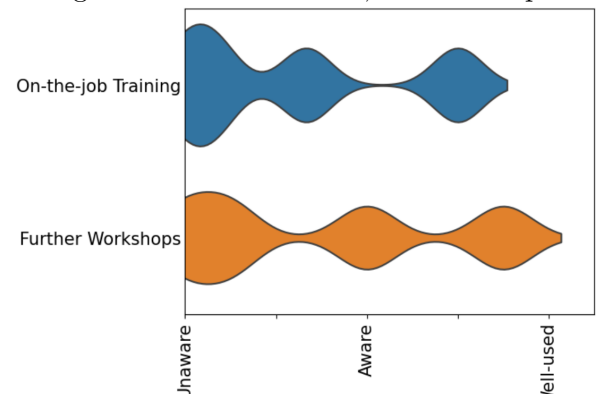
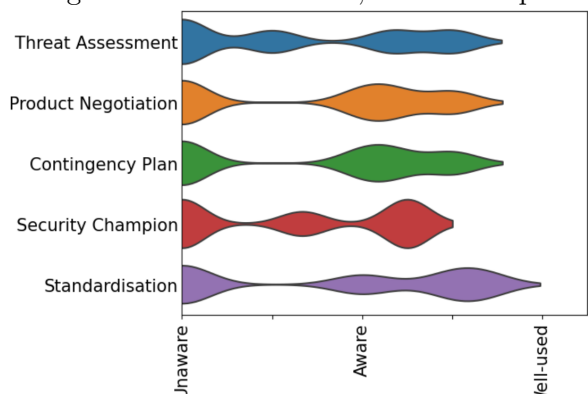
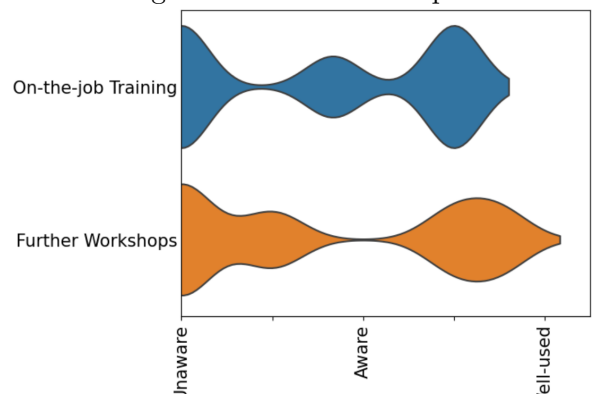
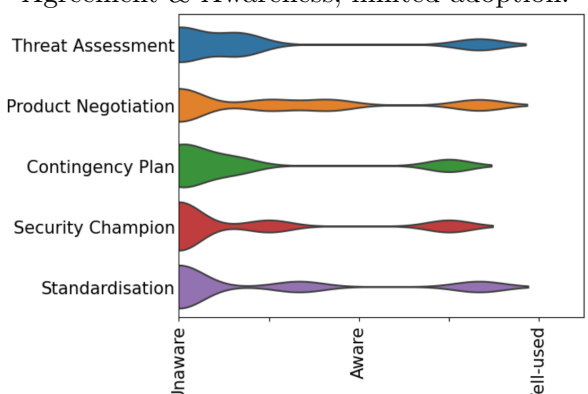
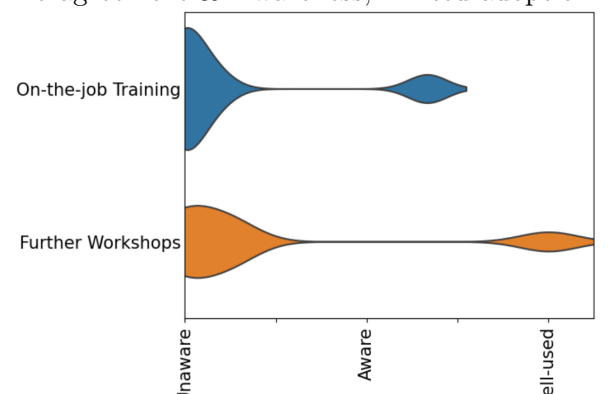
Team alias	Security importance	Problem finding techniques
Crocodile	<p>Security importance: 5; Priority range: 4 to 5</p> 	<p>Surprising agreement &amp; Limited awareness:</p> 
Hammers	<p>Security importance: 1; Priority range: 4 to 7</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 
Iron man	<p>Security importance: 7; Priority range: 1 to 5</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 
Komodo	<p>Security importance: 6; Priority range: 1 to 5</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 

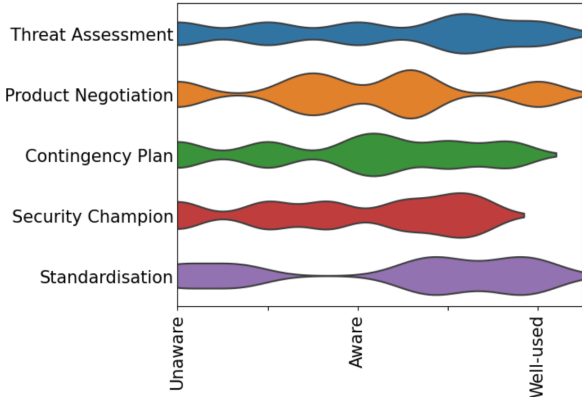
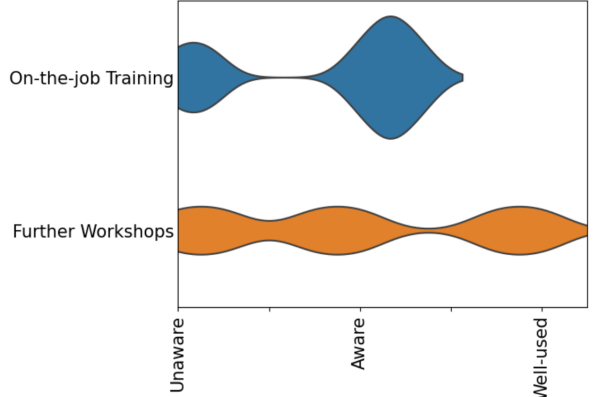
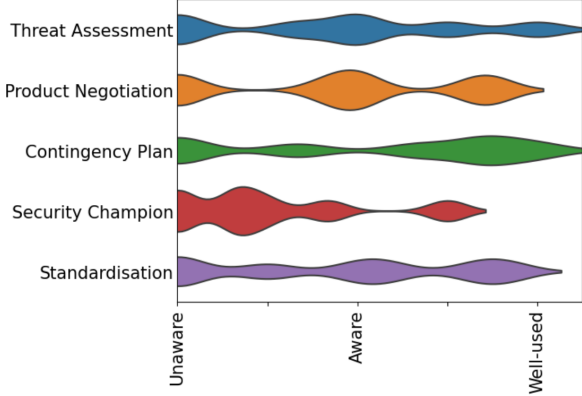
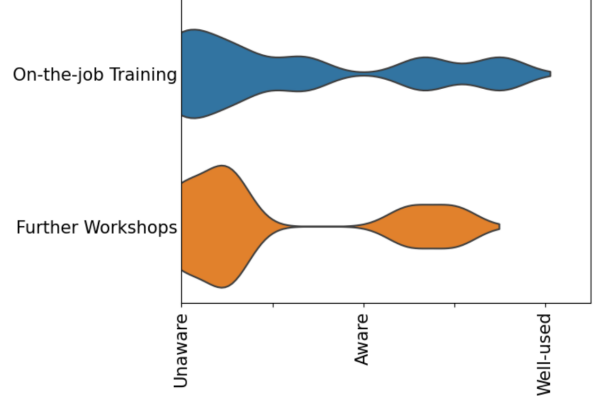
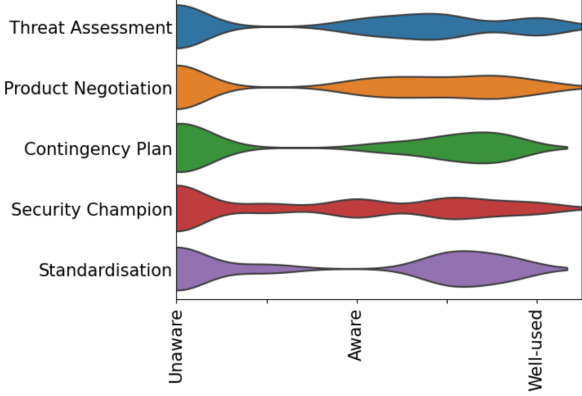
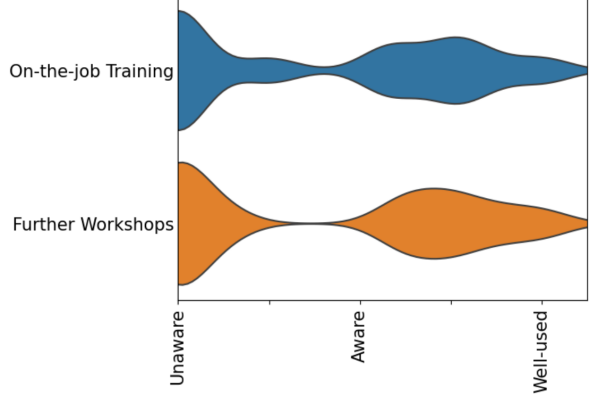
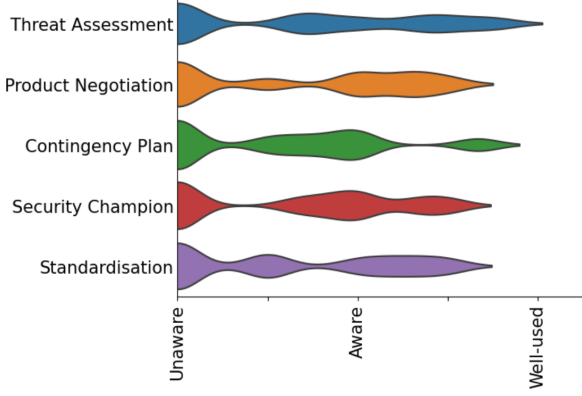
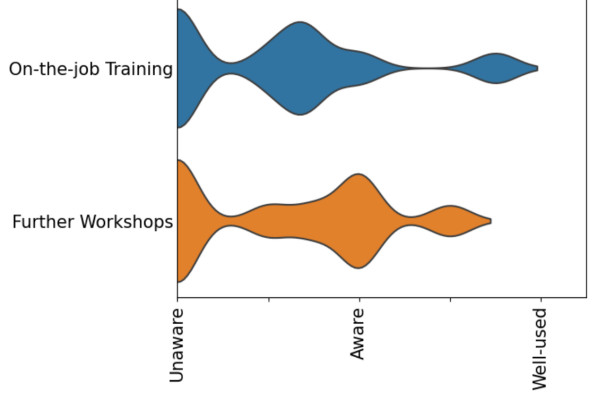
Team alias	Security importance	Problem finding techniques
Lizard	Security importance: 4; Priority range: 1 to 7 	No agreement & Some adoption: 
Mercury	Security importance: 5; Priority range: 1 to 6 	Agreement & Awareness, limited adoption: 
Phoenix	Security importance: 6; Priority range: 1 to 7 	Agreement & Awareness, limited adoption: 
Red Bulls	Security importance: 7; Priority range: 1 to 7 	Agreement & Awareness, limited adoption: 
Red Carpet	N/a	N/a

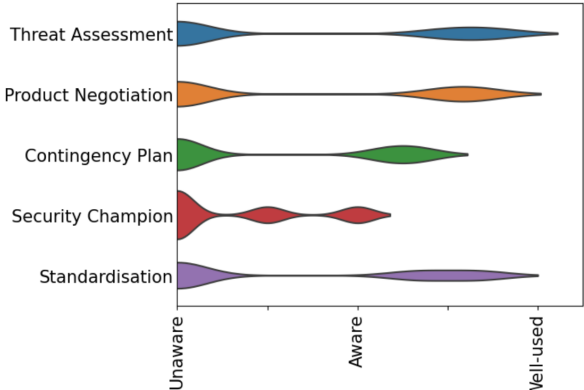
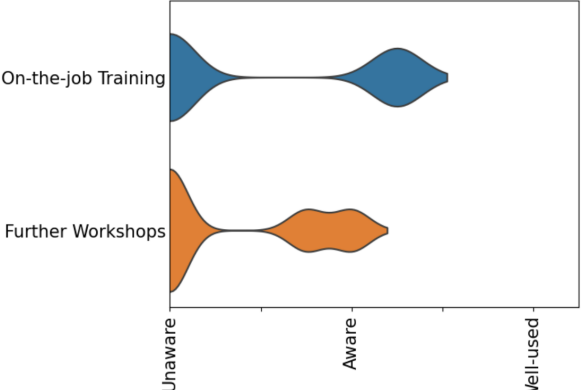
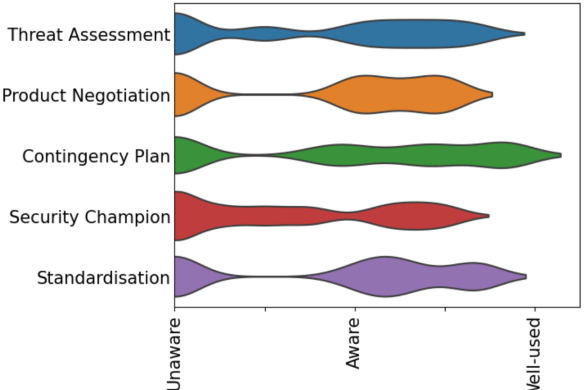
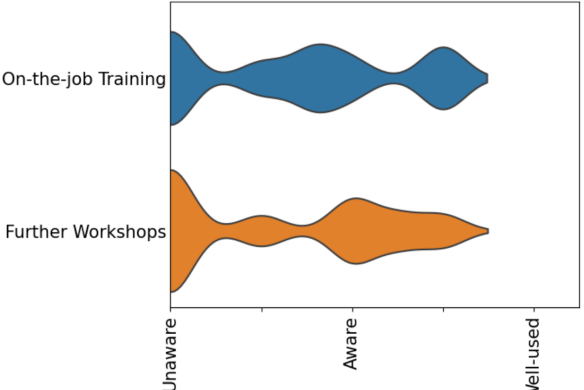
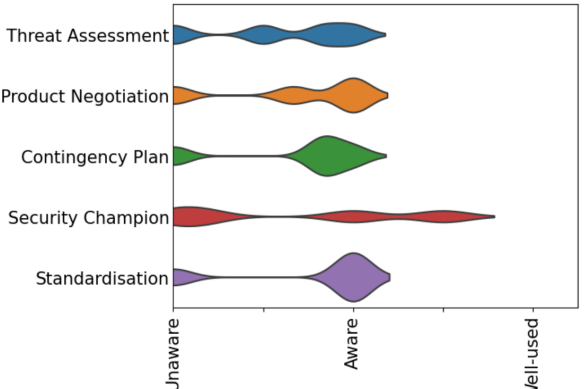
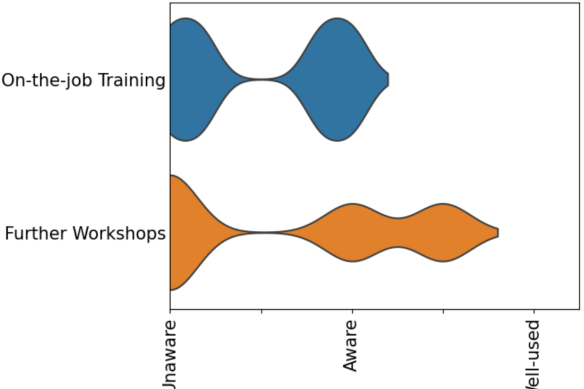
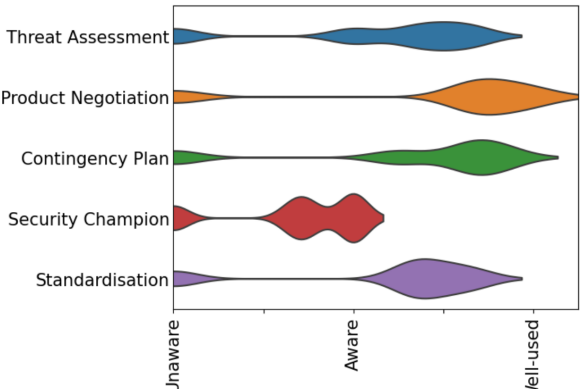
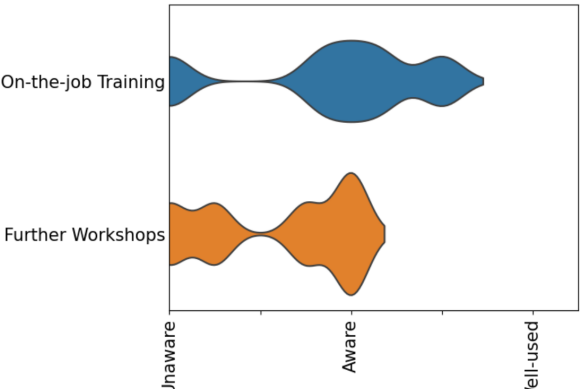
Team alias	Security importance	Problem finding techniques
Scissors	Security importance: 7; Priority range: 1 to 4 	No agreement & Awareness, limited adoption: 
Shield	Security importance: 7; Priority range: 1 to 3 	No agreement & Awareness, limited adoption: 
Shifu	Security importance: 3; Priority range: 1 to 7 	Agreement & Awareness, limited adoption: 
Spiderman	Security importance: 4; Priority range: 1 to 7 	Agreement & Awareness, limited adoption: 

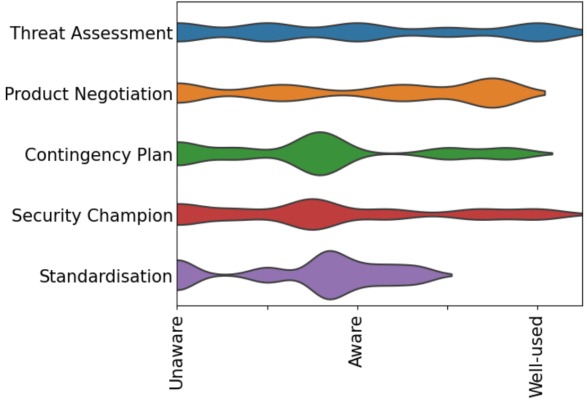
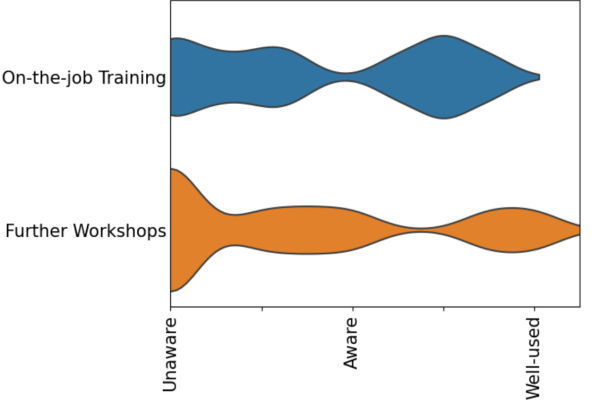


Team alias	Security importance	Problem finding techniques																												
Turtle	<p>Security importance: 6; Priority range: 1 to 6</p>  <table border="1" data-bbox="287 369 877 705"> <caption>Security Importance Data</caption> <thead> <tr> <th>Attribute</th> <th>Rating</th> </tr> </thead> <tbody> <tr> <td>Responsive</td> <td>3</td> </tr> <tr> <td>Scalable</td> <td>2</td> </tr> <tr> <td>Privacy respecting</td> <td>2</td> </tr> <tr> <td>Available</td> <td>1</td> </tr> <tr> <td>Features</td> <td>2</td> </tr> <tr> <td>Secure</td> <td>1</td> </tr> <tr> <td>Easy to use</td> <td>5</td> </tr> </tbody> </table>	Attribute	Rating	Responsive	3	Scalable	2	Privacy respecting	2	Available	1	Features	2	Secure	1	Easy to use	5	<p>Agreement &amp; Awareness, limited adoption:</p>  <table border="1" data-bbox="901 347 1500 705"> <caption>Problem Finding Techniques Awareness</caption> <thead> <tr> <th>Technique</th> <th>Awareness Level</th> </tr> </thead> <tbody> <tr> <td>Automated Pen Testing</td> <td>Unaware</td> </tr> <tr> <td>Automatic Static Analysis</td> <td>Unaware</td> </tr> <tr> <td>Configuration Review</td> <td>Aware</td> </tr> <tr> <td>Code Review</td> <td>Aware</td> </tr> <tr> <td>Pen Testing</td> <td>Unaware</td> </tr> </tbody> </table>	Technique	Awareness Level	Automated Pen Testing	Unaware	Automatic Static Analysis	Unaware	Configuration Review	Aware	Code Review	Aware	Pen Testing	Unaware
Attribute	Rating																													
Responsive	3																													
Scalable	2																													
Privacy respecting	2																													
Available	1																													
Features	2																													
Secure	1																													
Easy to use	5																													
Technique	Awareness Level																													
Automated Pen Testing	Unaware																													
Automatic Static Analysis	Unaware																													
Configuration Review	Aware																													
Code Review	Aware																													
Pen Testing	Unaware																													

Team alias	Process improvement techniques	Education techniques
Crocodile	<p>No agreement &amp; Some adoption:</p> 	<p>No agreement &amp; Awareness, limited adoption:</p> 
Hammers	<p>Agreement &amp; Some adoption:</p> 	<p>No agreement &amp; Awareness, limited adoption:</p> 
Iron man	<p>No agreement &amp; Awareness, limited adoption:</p> 	<p>No agreement &amp; Some adoption::</p> 
Komodo	<p>Agreement &amp; Awareness, limited adoption:</p> 	<p>No agreement &amp; Awareness, limited adoption:</p> 

Team alias	Process improvement techniques	Education techniques
Lizard	<p>No agreement &amp; Some adoption:</p> 	<p>No agreement &amp; Some adoption::</p> 
Mercury	<p>No agreement &amp; Some adoption:</p> 	<p>No agreement &amp; Awareness, limited adoption:</p> 
Phoenix	<p>No agreement &amp; Some adoption:</p> 	<p>No agreement &amp; Some adoption:</p> 
Red Bulls	<p>Agreement &amp; Awareness, limited adoption:</p> 	<p>Agreement &amp; Awareness, limited adoption::</p> 
Red Carpet	N/a	N/a

Team alias	Process improvement techniques	Education techniques
Scissors	<p>No agreement &amp; Awareness, limited adoption:</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 
Shield	<p>No agreement &amp; Some adoption:</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 
Shifu	<p>Agreement &amp; Awareness, limited adoption:</p> 	<p>Agreement &amp; Awareness, limited adoption:</p> 
Spiderman	<p>No agreement &amp; Some adoption:</p> 	<p>Agreement &amp; Some adoption:</p> 

Team alias	Process improvement techniques	Education techniques
Turtle	<p data-bbox="363 320 790 353">No agreement &amp; Some adoption:</p>  <p>The violin plot displays the distribution of awareness levels for five process improvement techniques. The x-axis is labeled 'Unaware', 'Aware', and 'Well-used'. The y-axis lists the techniques: Threat Assessment (blue), Product Negotiation (orange), Contingency Plan (green), Security Champion (red), and Standardisation (purple). Threat Assessment shows a wide distribution across all three levels. Product Negotiation and Contingency Plan show a concentration in the 'Aware' category. Security Champion shows a concentration in the 'Well-used' category. Standardisation shows a concentration in the 'Aware' category.</p>	<p data-bbox="895 320 1503 353">No agreement &amp; Awareness, limited adoption:</p>  <p>The violin plot displays the distribution of awareness levels for two education techniques. The x-axis is labeled 'Unaware', 'Aware', and 'Well-used'. The y-axis lists the techniques: On-the-job Training (blue) and Further Workshops (orange). On-the-job Training shows a concentration in the 'Well-used' category. Further Workshops shows a concentration in the 'Unaware' category.</p>

## Appendix D

# Template for evaluation interviews after applying Weir's Team Security Assessment

In this appendix, we present the document that we used to guide our evaluation interview sessions used to validate our security maturity results, and to evaluate Weir's Team Security Assessment [86]. The interview sessions have been conducted with two mobile application developers and a chapter lead, as discussed in section 8.3. This document contains some demographic information, various open questions and a few 5-point Likert scale questions.

# Evaluation interview after applying Weir's Team Security Assessment

## Radboud University - Master Thesis - ABN AMRO

<b>CANDIDATE NAME:</b>		<b>CONDUCTED BY:</b>	Onno de Gouw
<b>POSITION:</b>	<b>SPIKE:</b>	<b>POSITION:</b>	Intern
<b>INTERVIEW DATE:</b>		<b>START TIME:</b>	XX:XX
		<b>END TIME:</b>	XX:XX

What does DevSecOps mean for you? Do you consider this to be clearly defined by ABN AMRO?	
What do you think of the results that have been produced by your / the DevOps team(s)? Do they match your feelings?	
What is your overall opinion about Weir's Team Security Assessment? Do you have any pros and cons?	
Would you have any suggestions for improvement when considering Weir's Team Security Assessment?	
Do you think the generated results have an influence on your / the team's secure development practices? Why (not)?	
Does the number of security activities assessed influence your motivation to carry out secure development practices? Why (not) and when would you feel motivated?	
What do you think of Weir's Team Security Assessment's scope? Does it cover the most relevant areas of concern?	

	Very poor	Poor	Acceptable	Good	Very good
How would you rate the usefulness of the report resulting from Weir's Team Security Assessment?					
<b>COMMENTS:</b>					
How would you rate the questionnaire used for Weir's Team Security Assessment?					
<b>COMMENTS:</b>					
How would you rate the duration it takes to fill out Weir's Team Security Assessment?					
<b>COMMENTS:</b>					

**COMMENTS** | Any additional comments.