jamk.fi

A framework for securing internal business-critical infrastructure services

A structured approach for reducing systemic security gaps

Tero Hannula

Master's thesis
December 2018
School of Technology
Institute of Information Technology
Master Degree Programme in Information Technology, Cyber Security

jamk.fi

Description

Author(s)	Type of publication	Date
Hannula, Tero	Master's thesis	December 2018
		Language of publication: English
	Number of pages: 95	Permission for web publication: x

Title of publication

A framework for securing internal business-critical infrastructure services

A structured approach for reducing systemic security gaps

Degree programme

Degree Programme in Cyber Security

Supervisor(s)

Hautamäki, Jari

Assigned by

L M Ericsson

Supervisor: Ruohomaa, Sini

Abstract

The objective of the research was to develop a framework for protecting business-critical digital infrastructures and services by ensuring adequate coverage of defence and avoiding systemic security problems during implementation. The framework can support teams responsible for a digital infrastructure and improve their capability to identify and deal with threats in an autonomous fashion. The framework can also be used as a supplementing tool to help understand and deal with threats when implementing an information security standard or another security framework.

The framework collects threats into nine classes, covering areas from physical security to organizational issues. The threats were collected in an internal brainstorming session, from ENISA cyber threat taxonomy, and from Lockheed Martin's Cyber Kill Chain attack model. In addition, three targets were created for understanding the threat environment, human labour and system functionality. The targets contain controls and other defences linked to goals that define the desired long-term security outcomes, forming a clear three-layer structure for the framework and securing digital infrastructures. It can then be implemented to meet specific needs of each organization.

The coverage of the framework was assessed by benchmarking control lists of ISO 27001 and CIS controls against the areas in this framework. The benchmarking showed that the references do not cover the areas in this framework completely, and in some areas the coverage is almost non-existent. Examples of these areas are accidental human errors and issues in supply chain security. These areas have played a significant role in major breaches during the recent years. Implementing the framework could help to identify the problems in these areas and resolve them.

Keywords/tags (subjects)

Cybersecurity, business-critical infrastructure, systemic security

Miscellaneous (Confidential information)

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Hannula, Tero	Opinnäytetyö, ylempi AMK	Joulukuu 2018
	Sivumäärä 95	Julkaisun kieli Englanti
		Verkkojulkaisulupa myönnetty: x

Työn nimi

A framework for securing internal business-critical infrastructure services

A structured approach for reducing systemic security gaps

Tutkinto-ohjelma

Degree Programme in Cyber Security

Työn ohjaaja(t)

Hautamäki, Jari

Toimeksiantaja(t)

L M Ericsson

Ohjaaja: Ruohomaa, Sini

Tiivistelmä

Tutkimuksessa kehitettiin liiketoimintakriittisen digitaalisen infrastruktuurin palvelujen suojaamisen apuvälineeksi kehys, jolla voidaan varmistaa puolustuksen riittävä kattavuus sekä vähentää puolustukseen liittyviä systeemisiä turvallisuusongelmia. Kehys soveltuu käytettäväksi jonkun tietoturvastandardin tai -kehyksen apuvälineenä auttamaan uhkaympäristön ymmärtämistä ja suojaustoimenpiteiden toteutusta, mutta myös tukee infrastruktuurista vastaavan tiimin itsenäistä kykyä tunnistaa ja torjua uhkia.

Kehys jakaa uhat yhdeksään luokkaan, jotka koottiin opinnäytetyöprojektin sisäisen ideoinnin lisäksi ENISAn uhkaluokituksesta sekä Lockheed Martinin kyberhyökkäysmallista. Saatu luokitus kattaa alueet fyysisestä turvallisuudesta organisaatiotason vaatimuksiin asti. Näihin luokkiin liittyvät toimenpiteet jaettiin lisäksi kolmeen toimenpideluokkaan, jotka kohdistuvat ympäristön ymmärtämiseen, ihmisten työskentelyyn ja järjestelmien toimintaan. Näiden luokkien sisältö on joukko järjestelmän suojaamiseen tarvittavia kontrolleja ja muita puolustusmenetelmiä, jotka on sidottu ennalta määriteltyihin tavoitteisiin. Näin koko kehys ja sen viitoittama lähestymistapa saa selkeän kolmikerroksisen rakenteen ja se voidaan suunnitella suojattavan kohteen tarpeiden mukaiseksi.

Kehyksen kattavuutta arvioitiin kokeellisesti vertaamalla sitä ISO 27001 -tietoturvastandardin ja CIS:n kontrolliluetteloihin. Vertailussa havaittiin, että verrokit eivät kattaneet kehyksen määrittelemiä alueita kokonaan, ja osittain kattavuus puuttui lähes täysin. Näihin kuuluu esimerkiksi ihmisen tekemien tahattomien virheiden sekä toimitusketjuun liittyvien turvallisuusongelmien huomioiminen. Näihin liittyvät ongelmat ovat olleet keskeisiä viime vuosina nähdyissä suurissa tietovuodoissa. Kehyksen soveltaminen voisi siis auttaa korjaamaan järjestelmän suojaamiseen liittyviä puutteita myös näiltä osin.

Avainsanat (asiasanat)

Kyberturvallisuus, liiketoimintakriittinen infrastruktuuri, systeeminen turvallisuus

Muut tiedot

Acknowledgments

I would like to thank the following people for their support and insight that helped me in getting this work completed:

Sini Ruohomaa, PhD, my instructor during the effort for her endless stream of insight on various topics.

K. Patrik Wheeler from BNP Paribas Fortis for input on pain points in critical infrastructure cyber security.

Lauri Säisä from Ericsson for valuable insight on the organizational dimension of security risk management.

Leena Norros, professor (ret.) for our discussions that encouraged me to think about human factors viewpoint in digital security.

And finally, and most importantly, my family members Taru, Rianna (8 yrs) and Niklas (3 yrs) for understanding why I had to take the time from you.

Contents

Figures

Tables

1 Introduction

A secure and robust digital infrastructure is becoming an essential part for running successful business in digitalizing world. Better reliability and dependability in organizations' digital infrastructure is needed to keep the business running while minimizing unexpected outages and other undesired events that could affect other business functions, customers and stakeholders.

Business-critical infrastructure can be a compelling target for attacks, often having elevated potential for sensitive data leaks, deliberate service outages and other damage. The direct and indirect losses, both tangible and intangible, have potential to become extensive. We have seen this type of high-profile breaches appearing in the news feeds. Having control and understanding over the complexity of the network of systems, applications, their weaknesses, and understanding the potential threats is a difficult problem.

The root causes of problems are said to be *systemic* when they are not linked to individual persons or individual non-interrelated events, but to the way the system, the organization or a group responsible for something, works. Systemic problems generate opportunities for failures over time, so fixing these problems is important step in reducing long-term exposure for adverse events.

Managing security is easier for organisations that have personnel for the tasks. These organisations may use existing security standards and frameworks as a reference. However, there are also flat organisations with self-organising teams or highly specialised teams inside organisations that run their own infrastructure as a part of their specialized IT operations. These teams could benefit from a security framework that defines a methodology and language for understanding their threat landscape. The framework would then help in avoiding unintentional security failures. This thesis examines construction of an initial version of such a framework.

This thesis studies a structured approach to security. Example definitions for ”structured” in English dictionaries are "organized so that the parts relate well to each

other"[1] and "having a well-defined structure or organization"[2]. In this thesis we study adding structure to help in reducing dark corners and systemic problems in security.

1.1 Structure of the thesis

This thesis contains 7 chapters. In chapter 2 we set the research questions, select the research method and describe the research process.

In chapter 3 we describe the background of the topic with references to ISO 27000, NIST framework for improving cyber security and threat modelling.

In chapter 4 we construct a taxonomy for the framework with descriptions and examples on failures. The classification forms one of the two main parts of the framework.

In chapter 5 we construct the framework by adding the targets, which is the second main part of the framework.

In chapter 6 we benchmark the framework against ISO 27001 Annex A and CIS controls as an example to see how they cover the classes presented by this framework.

Finally, we make the conclusions in chapter 7.

[1] The Cambridge Advanced Learner's Dictionary
[2] The American Heritage Dictionary of English language

2 Research problems and methods

2.1 Research problem to be studied

Keeping a digital infrastructure secure is a challenging task because of overwhelming number of ways the security can fail. Also, the challenge of maintaining adequate level of security is a continuous task while attacks or accidental errors possibly leading to a breaches can be single occurrences. Breaches have potential of causing extensive consequences to the business, and the breaches may go undetected for an extended period of time.

This thesis is an initial and experimental effort of forming a security framework that could support a team of security practitioners and other related staff in identifying and dealing the threat environment while working with critical digital infrastructures and their services within an organization. The intended audience can be a flat organisation like a team of peers working mostly in an autonomous and agile fashion in product development and operations or similar. However, the framework can be useful for a traditional organisation where security administration is responsible for the security efforts, possibly derived from an security standard while possibly aspiring to conform to the standard.

One argument of this thesis is that having a dedicated and relatively simple framework can help in finding and reducing "dark corners" in security of digital business-critical infrastructure by giving directions on what areas need attention to understand the likely threats and how the task should be approached.

There are four main research questions to be studied:

Q1. What are the areas that need most attention while securing business-critical digital infrastructures?

Q2. How could a team be directed in its effort to secure these areas with assistance by a practical framework?

Q3. Does the new framework provide enough coverage for the identified topics?

Q4. How this framework could be applied in practice?

The framework is built to provide a holistic view to the security at the operational team level, covering various topics from physical security to human and teamwork related issues.

2.2 Research method

The goal of this thesis is to construct a new framework for the aforementioned purpose, so we follow the principles of constructive research throughout the process. Ojasalo *et al.* support this decision by stating that constructive research is relevant for producing a new concrete product, plan, model, framework, or similar (Ojasalo, Moilanen ja Ritalahti 2014).

The purpose of the framework is to provide a simple tool for a team or a flat organisation to improve security of their digital infrastructure. This work may end with an innovation, although the project is experimental with a lot of uncertainty on what the outcome will be. This is not a problem from the research methodology viewpoint, since according to Ojasalo *et al*., producing innovations is unimportant in constructive research, while the emphasize is in use of theoretical information and conceptual design to produce something that is new and practical.

The goal, however, is to create something that could improve security activities in the given working environments where existing frameworks may be regarded as difficult to approach. According to Kananen, this conforms to goals of constructive research since constructive research is regarded as one form of intervention research where the goal is changing something for the better by solving a problem instead of explaining and understanding something (Kananen 2017). This is further supported by Kasanen *et al.*, who state that the goal in constructive research is to improve activities and procedures within an organization. They also note that the difference between constructive research and other forms of construction is in usage of theoretical background (Kasanen, Lukka and Siitonen 1993). For this thesis, background information is collected from two existing standards and framework and research material to help founding the framework construction.

2.3 Research process

Research process for constructive research contains the following steps (Kasanen, Lukka and Siitonen 1993):

P1. Find a practically relevant problem which also has research potential.

P2. Obtain a general and comprehensive understanding of the topic.

P3. Innovate, i.e., construct a solution idea.

P4. Demonstrate that the solution works.

P5. Show the theoretical connections and the research contribution of the solution concept.

P6. Examine the scope of applicability of the solution.

With the research goal and method selected, these steps are implemented with one adjustment in the testing phase as explained next.

The work proceeds in four phases. The first phase of the research process uses brainstorming session to collect initial information from various levels that need to be addressed in our framework. The results will give the initial directions for the threat taxonomy formation.

The second phase of the process gathers more information is gathered from different sources like research papers, standards, frameworks, guidelines, books and blog posts to find indications of importance that should be addressed by our framework. The standards and guidelines in line with this topic are the *ISO 27000* and *NIST Framework for Improving Critical Infrastructure Cybersecurity*.

In the third phase, the ideas are classified in two runs: first by collecting them into a number of classes, and then defining how to approach the classes. Classes will tell the areas in security that need to be looked for. In addition, a structure in how the identified areas are dealt with will be defined. The result of this phase is the framework.

In the fourth phase, the framework is benchmarked by comparing its coverage to find similarities and differences against the ISO 27000 series and CIS controls. This is a simplified form of testing of the framework to understand its viability and applicability and a planned deviation from the research process as described earlier by

Kasanen *et al*. Because of limited resources available for this thesis, we cannot do an implementation of the framework. Proving that the framework "works" would be a major task on its own and is therefore left for another effort.

To have a practical view on the topic, an example case of an imaginary company and its business-critical infrastructure is carried throughout the construction process to have something to reflect the concepts on.

3 Motivation and related work

This thesis is about developing an initial version of a framework to support a mostly autonomous and agile IT team in their need to deal with their threat model while running an internal business-critical infrastructure and keep it secure.

Resulting from business impact assessment, a part of IT may be identified as a business-critical asset with need to keep it secured accordingly. To protect the infrastructure, the realistic threats need to be understood before defensive actions. This part of security activities could be helped by a relatively simple reference framework that improves understanding the threats and create a common language on the subject how to deal with the threats. Security related tasks could then be more committed and effective among the team of practitioners who are working with that part of the IT on daily basis and with those who administrate the security work.

The difference in focus and required amount of work can be a reason why implementing the aforementioned standards and frameworks may not be the desired way to secure the infrastructure. The team may find too little practical support for the effort. Implementing the ISO 27000 for a team inside an organisation was studied by Lasse Laukka in his thesis on implementing ISO 27000 for a CERT team (Laukka 2015). He found the effort possible but rather tedious with a lot of documentation mandated by the standard. The most obvious difficulties were generality of the standard requiring a lot of time and effort to complete, room for misinterpretation of the control objectives and controls, difficulty in defining the scope for the information security management system (ISMS) without prior experience on the standard, and risks in requirement to certify motivating the process instead of doing effective risk mitigation (Laukka 2015, 41). These findings suggest that the ISO 27000 may not be the obvious tool for cases that are in the focus of this thesis. Moreover, our goal is not to create an ISMS but a simple security framework to reduce the amount of gaps and dark corners in threat identification, and then provide better basis for risk assessment and risk mitigation that are outside the scope of this thesis. This framework can also be used as a complementary tool when implementing an ISMS or some other security framework.

We next construct and example case that is used to reflect the ideas behind the the new framework. Then we examine briefly two existing standards and frameworks for a reference.

3.1 MediMaze, an example company with business-critical infrastructure

To reflect security issues from different threat classes to a common case, we construct an imaginary organization with a business-critical infrastructure that needs to be protected.

MediMaze, a company specialized in producing digital medical devices for hospitals around the world. The product range covers X-ray devices, CT scanners, MRI devices, medical ventilators and others. MediMaze has outsourced its production of physical components and assembly lines to several subcontractors. All the software is developed in-house. The subcontractors install the software into the devices.

Customer devices are connected to a hospital network that has access to MediMaze's network. Devices have configuration for connectivity information and their digital identity. MediMaze keeps track of each device's status in terms of the device's origin, validity of the annual license, firmware version, validity of the firmware, current patch level and others.

MediMaze provides software updates to its devices for features upgrades according to each client's subscription plan that may change over time, and software patches to fix identified defects.

The medical devices poll the MediMaze servers periodically to learn about the subscription data and the lease agreement. When a lease period is approaching its end, the staff of the customer organization starts receiving notifications about the issue. If the lease period ends, the device goes to a safe minimum working mode where the device is less useful for the customer but still safe enough for patients.

MediMaze has an office with an internal network for developers' workstations and for other employees' computers, a wireless access point and a network printer. There is also a backend network behind a firewall that has all the data of the medical

devices in production use, their private keys and applications for the functions described earlier. This part of the infrastructure needs to be up all the time with no outages. Loss of keys and data would lead to loss of control over the medical devices produced, or an attacker changing the devices' configuration and taking control over them. These events would be a risk to patients' health and an intolerable loss to MediMaze's business. Therefore this part of the infrastructure is considered a business-critical asset. Also, all the medical devices in production use can be regarded as business-critical infrastructure because of their importance to support human life, and failures beyond minor issues and nuisances cannot be accepted. A wide-spread attack on these devices rendering them useless would be a disaster.

3.2 Relationship to other related methods and standards

In this chapter we look briefly in other methods, standards and frameworks that are referred to in this thesis and is useful in understanding this framework's position in relation to them and how the framework is different.

3.2.1 Relationship to ISO 27000 standard

ISO 27000 (ISO/IEC 2009) is a family of international standards that define an information security management system (ISMS). The purpose of the standard is to provide a model for establishing, implementing, operating, monitoring, reviewing, maintaining and improving the protection of information assets in an organisation.

The ISO 27000 standard family is extensive and has different focus in comparison to the framework presented in this thesis. The focus of the ISO 27000 is to provide a framework for organisation's management to build and maintain an end-to-end information security management system. It is a wider scope with management aspects like responsibility to lead the implementation and requirement to have all the required documentation done and maintained, and have roles set in the organisation. The standard does little in defining roles or activities for frontline practitioners.

The standard is divided in two main categories, the mandated standards and supplemental standards. The ISO 27001 is the centrepiece of the standards family and one of the two mandated parts. This standard defines how to create and maintain an ISMS for an organisation. This part contains a requirements part and a list of security

controls in the Annex A. The standard does not exclude use of additional control sources. The ISO 27002 provides an implementation guide in form of control objectives and best practices for implementing security controls, while the ISO 27003 provides further guidance for ISMS implementation. ISO 27004 is a use of measurements to assess effectiveness of the ISMS implementation. ISO 27005 provides guidance for a risk management process. ISO 27006 contains certification body requirements, and is the second mandated part of the standards family, directed for organizations certifying others for ISO 27001 compliance.

Our framework, however, has a more limited scope of empowering autonomous teams, mostly consisting of frontline practitioners, in understanding the changing threat landscape during daily operations, and how to defend against the threats before they cause incidents. As we will later learn, this is an attribute of organisations that can operate successfully under variety of threats and avoid paths to failures. This framework has limited coverage for digital security management aspects.

ISO 27000 standard is designed to be used in all kind of organisations. In case of small or flat organisations, the implementation requires adaptation to be feasible for the available resources. Still, there can be opportunities for security gaps that may go undetected. This framework can help in filling these gaps and help to achieve the security goals behind the decision to implement the ISO 27000 standard.

We will see later in chapter 6.3 how the coverage of controls in ISO 27001 Annex A compare to this framework when we benchmark the two against each other.

3.2.2 Relationship to NIST Framework for Improving Critical Infrastructure Cybersecurity

The NIST Framework for Improving Critical Infrastructure Cybersecurity (later “NIST framework”) is a voluntary risk-based security framework for putting together and communicating organisation's cyber security strategy (NIST-2 2018). It was prepared by the National Institute of Standards and Technology (NIST), a federal agency within United States Department of Commerce, to be used as a tool for securing critical infrastructures, which is defined as "*systems and assets, whether physical or virtual, so*

vital to the United States that the incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters."[3].

The framework does not limit its scope to critical infrastructures but encourages any sector or business regardless of its size to consider adopting it for cyber security efforts.

The NIST framework consists of three parts: Framework Core, the Framework Implementation Tiers and the Framework Profiles (NIST-2 2018, 6).

The Framework Core (NIST-2 2018, 6) contains key high level concurrent cyber security core functions: identify, protect, detect, respond and recover. These 5 functions are divided into classes, and the classes are divided further into subclasses. Each subclass is provided with a set of informative references to external documents that suggest useful controls for implementing requirements in that subclass. The structure of the Core functions is shown in Figure 1.

The Framework Tiers (NIST-2 2018, 8) describe four levels of sophistication for the selected security posture based on risks environment and business security requirements. The higher the cyber security requirements, the higher the selected Tier would be.

The Framework Profiles describe the outcomes of the security activities (NIST-2 2018, 11). The suggested method is to first create a profile to describe current security outcomes, and then create a target profile to identify opportunities for improved outcomes. These profiles can be used to communicate the security posture internally and also externally to business partners. The target profile is created by selecting controls for subclasses that align with risks and business requirements. The framework does not provide templates for creating profiles.

[3] USA Patriot Act of 2001 (42 U.S.C. 5195c(e))

Figure 1. Structure of the NIST cybersecurity framework Core functions (NIST-2 2018).

The NIST framework itself does not provide controls but encourage use of external control sources. The framework refers to several external control sources, one such source is CIS Controls maintained by Center for Internet Security (CIS)[4]. To see how the CIS controls compare to our framework, we will benchmark the two in chapter 6.4.

The goals in the NIST framework and our framework have commonalities in terms of securing digital infrastructures, but as we see from the definitions and descriptions, their focus, scope and approach are different. NIST framework is risk-based approach with a scope of protecting critical infrastructure in society, while our framework is more narrowly focused to deal with threats on the part of IT that was identified as business-critical in business impact assessment. The NIST framework is a management level tool for organisations while our framework may be easier to approach in flat organisations or mostly autonomous IT teams. The two frameworks are therefore complementary and may provide improved outcomes when used together.

[4] https://www.cisecurity.org/

3.2.3 Relationship to threat modelling

Threat modelling is an activity to find and address security problems by means of using models. Modelling abstracts away the details, which helps in asking more generic questions on what might go wrong. Shostack presents a four-step approach in his book, each step having its own methods (Shostack 2014):

- What are you building?
- What can go wrong with it once it's built?
- What should you do about those things that can go wrong?
- Did you do a decent job of analysis?

Threat modelling covers everything having security properties, so it is suitable for improving security of digital systems. Threat modelling itself does not define methods or how to structure its activities, it's up to the practitioners to decide using some criteria how threat modelling is done in each case. Threat modelling is likely to start for a specific reason at some point of time and last for a predefined timespan. Reasons to start threat modelling could be design of new software, introduction of a new server in a network, change of configuration, and others.

This framework has similar goals to thread modelling, identify potentials ways how security could fail and then deal with the identified threats. From that perspective, this thesis could be seen as an effort of forming a high-level threat modelling framework. That, however, would be a simplification. The result of this thesis is not a technical guidance on how to do threat modelling but a guidance on how to plan and arrange security related activities in daily operations of a business-critical infrastructure, with a goal of having less security gaps for systemic reasons that could keep forming opportunities for breaches.

4 A taxonomy of cyber security for business-critical infrastructures

Cyber security is a wide topic, providing a plethora of opportunities for both success and failure. To be successful, we need to understand what need to be protected, from what or whom and how. Better understanding of the variety of potential threats is essential in defending the assets against the threats.

We start the framework construction with a brainstorming session to have something to start with. The question is "what sources of threat a critical infrastructure could face?".

Then we continue by selecting an existing threat taxonomy as a base with a wide coverage. To keep things simple, an own taxonomy is derived from an existing one.

Then other sources are used to see if the classification is ready or to see if new classes are needed.

The result is a combination of these sources. The taxonomy is later used in building the framework. Instead of a comprehensive coverage for any organization, the classification is intended to cover areas in security that are likely to be a common baseline when securing digital business-critical infrastructures.The Fourth Chapters's

4.1 Defining classes and contexts

4.1.1 Input from internal brainstorming session

The first set of ideas was gathered in an internal brainstorming session where likely causes of security issues were collected on a whiteboard. The issues were then classified so that related issues were put in their corresponding class. The result of this session was not regarded as final but a starting point for the next session. The outcome was the first provisional version of the classification with 5 classes and another 5 groups as suggested classes. The result is shown in Appendix 1.

In the next phase the initial set of issues were examined further with some added ideas, and the result was the first version of the threat classification. The outcome is a classification with 8 classes as shown in Appendix 2. This classification received changes later during the process before becoming the the final classification.

During this session the issues were also looked from another viewpoint to identify how they might regard to actual activities in an organisation. This examination produced another classification of 3 classes which were later renamed as "targets". The outcome is shown in Appendix 3. This classification was left open for more ideas, but eventually the three targets remained the final version with improved definitions later in the process.

4.1.2 Input from ENISA threat taxonomy

There are many taxonomies created in the field of cyber security. One example is threat taxonomy by ENISA that was created for its internal collection and consolidation of threat information (ENISA-2 2016).

ENISA's taxonomy is extensive with 31 different threat taxonomies as its source. It classifies threats on their type, containing more than 100 classes as shown in Figure 2. This is good starting point, and obviously too complex and unnecessarily fine-grained for our purposes, so we derived a new more coarse-grained classification from it by combining parts of the brainstorming session.

The ENISA taxonomy could be translated to many kinds of other taxonomies. In our translation, we divided the items into two high level groups of security issues: classes and contexts. Each derived class identifies the direct source of a security issue, and therefore, what is likely the most effective point of control or resolution. Contexts identify the environment where the threat could realize.

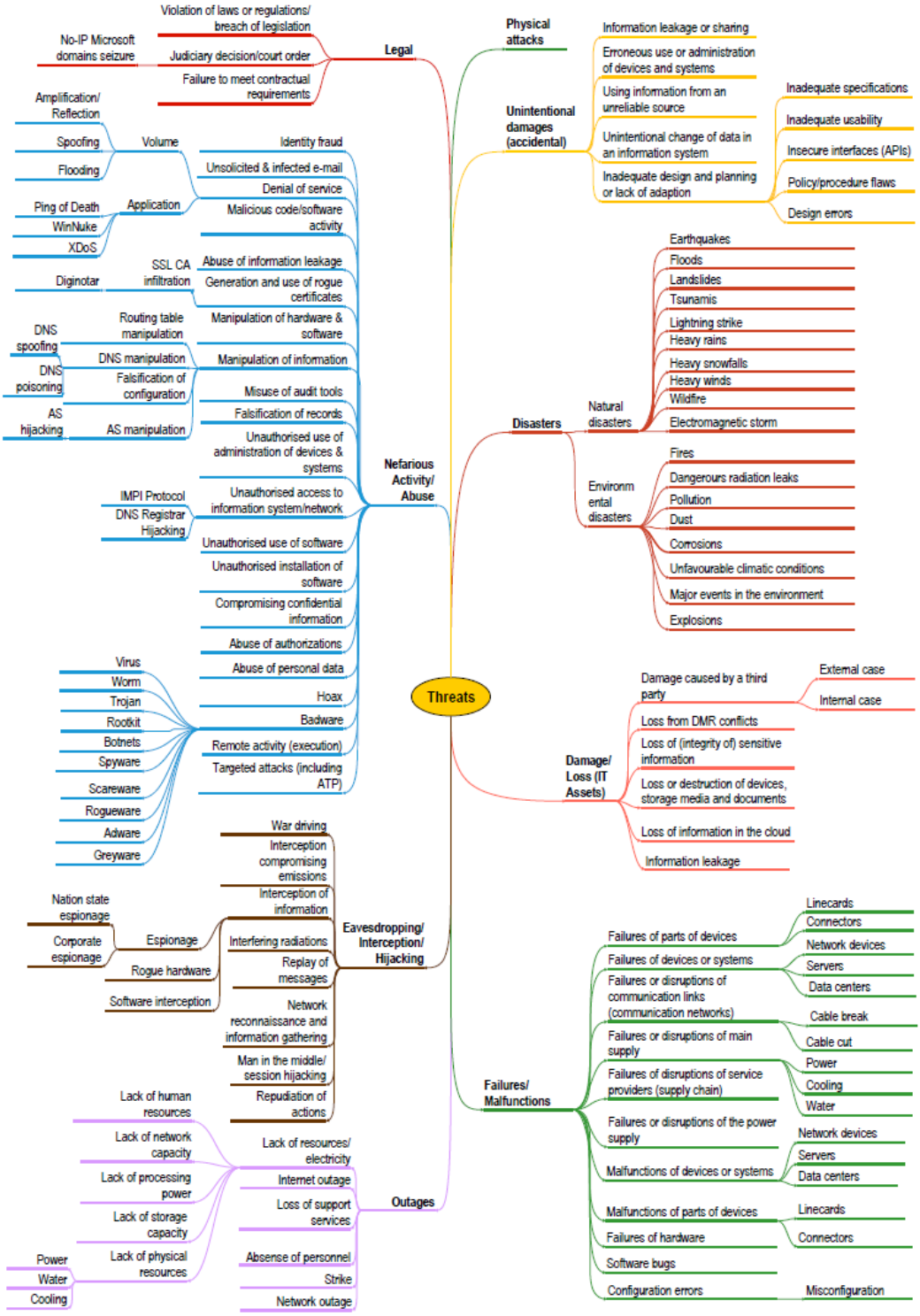

Figure 2. ENISA cyber threat taxonomy (ENISA-2 2016).

Each item in a class is likely to have a context. The decision on which class and context a security issue in the ENISA taxonomy should be put to in our framework can be determined by asking two questions:

1. What is the source of the security issue?
2. In what environment the issue emerges?

The answer to the first question suggests the class for the threat. The answer to the second question suggests the context for the threat. The correct answer may not be unambiguous, and in such cases the threat could be put in multiple classes or contexts. For example, a software defect in a network application that creates a vulnerability in a connected workstation would be classified as a software issue in a context of workstations and networks. It is not classified as a network security issue since that would be too high-level abstraction in our taxonomy, leaving the source of the threat unclear.

Each threat type from the ENISA taxonomy is placed in a group that has a clear origin. For example, all the classes in "Physical attacks" and "Disasters" are combined into "Physical" class since these threats are about physical access or physical action on an asset. Subclasses in "Failures/Malfunctions" are divided into "Software", "Hardware", "Configuration" and "Supply chain" depending on where the threats' origins are. This reclassification is done for all classes in the ENISA taxonomy.

4.1.3 Input from cyber kill chain

Another source for ideas is the "cyber kill chain", an attack model originally presented by Lockheed Martin (Lockheed Martin 2015). As there is a common structure in executing a military attack on a physical target, there is a common structure in executing a cyber attack on a digital target.

The model presents only one type of an intentional attack. Because this kind of attacks are common and likely more common when attacking business-critical infrastructures, the kill chain was selected as a source of data for our framework instead of a target for the final assessment. Other attack models have been

presented, like FireEye Attack Lifecycle[5], Gartner Cyber Attack Model[6] and MITRE ATT&CK Lifecycle[7].

The Lockheed Martin model describes the following seven phases of a cyber attack (Lockheed Martin 2015, 4). For each phase, we selected defences that could most likely contribute to the formation of the framework and left the defences with specialised skill like malware analysis outside of this examination.

1. Reconnaissance: the attacker is planning his attack and searches for the suitable point of entry into the target. Defence is restricting the outgoing information and understanding what the actor is trying to know. This information can be collected from logs. Another defence is detection for client behaviour that is specific to reconnaissance.
2. Weaponization: the attacker builds malware and exploits as a deliverable payload. Defensive action is finding information about ongoing malware campaigns since they tend to be reused.
3. Delivery: the attacker launches the attack by conveying the payload to the target. This can happen by delivering the malware by an email, social media, or a USB stick, or by attacking the target directly. Defensive actions include system hardening, keeping all the software updated, using appropriate tools like intrusion detection and prevention systems and application firewalls, and mainaining awareness of attack campaigns and their methods.
4. Exploitation: the vulnerability in the hardware, software of human is exploited to gain access to the target. Defensive actions are identifying malicious content delivery, use of secure coding techniques, regular vulnerability scanning and system hardening.
5. Installation: the attacker installs a backdoor to gain persistent access to the target. Defensive actions are use of endpoint security software like HIPS, auditing endpoint processes, and limiting user privileges to the minimum.
6. Command and control (CC): malware contacts the attacker for commands to manipulate the attacked system, typically over web, email or DNS. Defensive actions are blocking CC infrastructure through system hardening, use of proxies for CC protocols and finding information on new kind of CC infrastructures.
7. Actions on objectives: with access to the target, the attacker may try to move to other targets, escalate privileges, do damage and exfiltrate data. Defensive actions are detection of data exfiltration, incident response actions and network packet capture for analysis.

[5] http://www.iacpcybercenter.org/resource-center/what-is-cyber-crime/cyber-attack-lifecycle/
[6] https://blogs.gartner.com/ramon-krikken/2014/08/08/introducing-gartners-cyber-attack-chain-model/
[7] https://www.mitre.org/capabilities/cybersecurity/threat-based-defense

Taken together, the defences against the attacks conforming to characteristics of the cyber kill chain are understanding the threat environment, user awareness, secure system configuration, anomaly detection and secure coding practices. In addition, there are several defences that are outside of the scope of this framework like malware forensics.

This examination did not provide any new classes in the taxonomy, so we have a reason to believe that the taxonomy is free of obvious weaknesses.

4.1.4 The final taxonomy

Before putting all the information together, a class for interoperability security and supply chain security were added to get direct attention to the topic. Also, a class for human error and a class for security awareness and capability were defined to make a distinction between two issues: an error caused by individual humans trying to do their work but failing because of an accidental error, and a human error actively directed into failure by victimising them by an adversary or by their organisation by mismanaging the security or working culture. The existing frameworks and IT industry already deal with "user education" to a degree, without much depth in the nature of human error and how it can affect security.

The work eventually levelled off with 9 classes as shown in Figure 3. The classification appears comprehensive with no obvious gaps in how it covers areas of IT, each with their opportunities for making or breaking security. The taxonomy starts from physical security, then moves to technology starting from hardware, continuing to individual software and interaction between software, then proceeding to encryption and configuration of objects like software, systems and networks. After covering technology, the taxonomy goes to level of individual humans and then to their organisation, which could be a team of peers or something larger. There the taxonomy covers interaction with third parties and the organisation's awareness for security and capability to do related actions accordingly.

The taxonomy is presented as a set of discrete classes, however, in reality they overlap. For example, the class for human error is likely present in all other classes since everything in digital environments is made, used and maintained by humans.

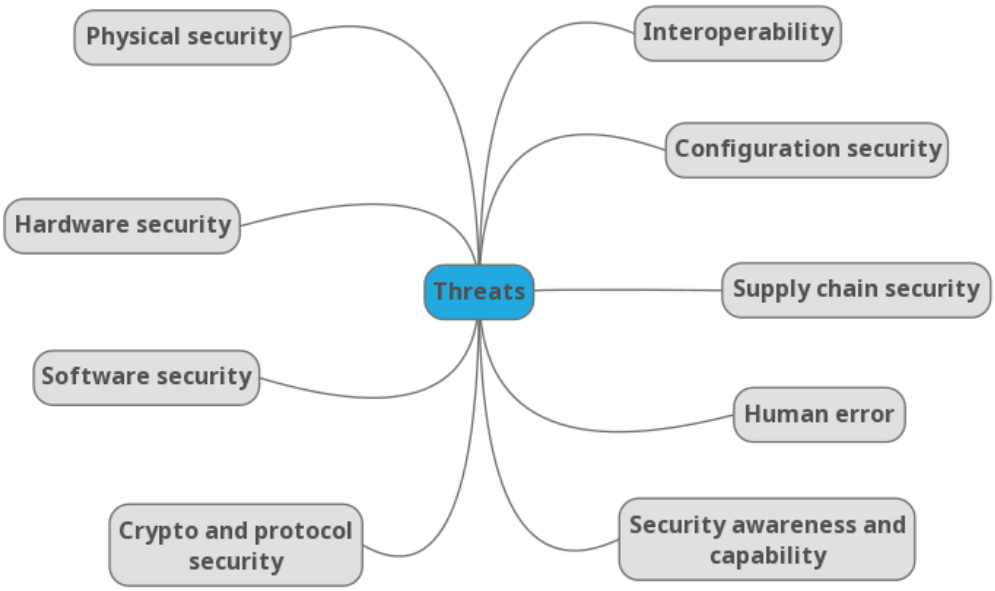

Figure 3. A simplified threat taxonomy after reclassification.

The classes are still without their definitions, and we will form them in the next section.

The purpose of having contexts was to understand the line between threats and their environments of appearance during the classification work. Networking software defect is a software defect, and having a class for network software threats are not likely to add value for the framework in terms of coverage or usefulness. We are not using the contexts from now on in this thesis.

The mapping from the ENISA taxonomy to our taxonomy is not "1-to-1", because the classes have different scopes. For example, class "network outage" in ENISA taxonomy has no one corresponding class in our taxonomy since a network can fail inside or outside of our organisation, and do it for many different reasons. Therefore this class could be translated to hardware security, software security, configuration security, human error and security awareness depending on the origin of the security

issue. The translation to "security awareness and capability" is appropriate for cases where the network outage happens outside of our organisation and the business needs to be recovered according to documented business continuity plan.

Similarly, class "loss of sensitive information" in ENISA taxonomy has no one-to-one mapping to our taxonomy because of missing information on what kind of information is lost and how. The same applies to "targeted attack including APT", which could be done in multiple ways and by combining multiple methods. This threat could be mapped to all classes in our taxonomy.

With the resulting 9 classes, we assume this as a minimum viable taxonomy that is complete enough to cover the needs of this framework. Proving this to be a correct assumption cannot be done within the resource limits of this thesis. We benchmark the converage of this taxonomy to ISO 27001 Annex A and CIS controls later in this thesis, which is expected to reveal any obvious shortcomings in the taxonomy.

To understand the scope of each class, we next describe each of them with some examples on how each of them have somehow failed recently in the real world. This examination provides a practical view on the problem of keeping protected assets safe. We'll see that all classes have seen security problems, and there is little reason to believe that these problems will end in the foreseeable future. This helps us understanding the need for the continuous work to keep the assets safe. Each class also has an example how vulnerabilities in the class would affect the example organization, MediMaze.

4.2 Physical security

4.2.1 Description

Physical security covers situations where the protected assets come to a physical contact with a threat actor and for that reason suffers a loss on integrity, availability or confidentiality. The actor can be a human manipulating, operating, damaging or stealing assets after either breaking in or after getting physical access to a secured area by tailgating, piggybacking or by getting access to a key card of an employee, or having an insider accessing the assets and causing a security event either accidentally or intentionally. The class contains also physical access to phased out and disposed

assets that still need protection. The actor can also be related to the physical premises, where uncontrolled fire, water spill, physical jolt, electric shock, magnetic field and gas leaks can have an effect on assets. The actor can be the infrastructure like having a temporary outage of electricity. The actor can also be of natural origins like weather or seismic events.

4.2.2 Physical security issues in practice

The increasing number of laptops, mobile devices and portable media expose employees and organizations to incidents where loss of a device can lead to loss of confidentiality of sensitive information. The ENISA Threat Landscape Report 2017 states that physical theft was the main source of breaches until 2017 when online hacking and malware superseded it from the top position (ENISA 2018, 68). However, the black market for stolen mobile phones has halved during 2009-2017, most likely because of improving security measures implemented by device vendors (68). Overall, physical actions were present in ca. 8 percent of all breaches.

Data can also leak by stealing a computer or its mass storage and then mounting the storage into another device. The ENISA report states that 70 percent of people have lost a data storage device, and 7,5 percent of people have lost their laptop in the last 12 months (69). The file synchronizing feature worsens the problem since files from multiple devices can be exposed when a person loses a single device.

4.2.3 Physical security in MediMaze

In our example case, the initial attack vector could be someone entering the premises by tailgating the employees through doors or breaking inside, or walking in because doors and locks do not work properly, or using an employee's key. The attacker could then steal equipment and documents, or use a USB device to read sensitive data from computers, or cause physical damage. The attacker could install devices in open network interfaces to listen traffic for valuable information like access credentials, or modify the data in transit, or relay the data to the attacker over a wireless link and below perimeter security defences. The attacker could reset the passwords in computers by using appropriate tools and then log in to read the data. The company laptops and mass storages could be stolen or lost, leading to loss of

equipment and possibly loss of sensitive data. Data can also leak when disposed documents or devices containing sensitive data is available in a dumpster or recycling area.

All these events can also prepare attackers for escalating the attack to affect services in the business-critical infrastructure.

Physical damage and possibly loss of availability of the critical infrastructure services could occur when electricity supply or data communication is cut as a result of, for example, maintenance work inside or outside of the premises. The premises can also suffer damage by a broken water pipe leaking water inside the premises, endangering equipment.

4.3 Hardware security

4.3.1 Description

In this thesis we define hardware security as properties of physical devices capable of causing security problems. Hardware security covers devices like a hard disk drives, CPUs, copy machines, network printers, routers, wireless keyboards and HSM modules. Hardware runs all the software that handles and stores data and controls systems and is an important link in security. A hardware failure is a likely source for loss of availability, but they can also be sources of problems in confidentiality and integrity. Hardware devices have a relatively narrow scope and may run a dedicated piece of software that is typically updated as one firmware file or cannot be updated at all. In some instances like CPUs the line between software and hardware can be difficult to judge since parts of the internals are run by software, or microcode in CPUs, that may not manifest itself to outside.

Firmware update may require manual work on each device to have the firmware updated instead of have to process automated as a part of centralised patch management. This property also makes it more laborious to test the new firmware for correct operation, possibly requiring a second set of hardware. All the extra effort, cost and trouble may lead to a situation where hardware security gets outside of the focus and eventually causes security issues. For these reasons, hardware security needs special attention and deserves to be separated from software security for cases

where software is involved. Issues related to computers are not covered in hardware security since computers are typically compositions of multiple hardware devices and are used as generic computer software execution platforms with changing set of arbitrary software. Computers are or can be updated as a part of centralised and automated patch management.

4.3.2 Hardware security issues in practice

In 2017, a defect in Infineon's RSA library was found to generate key pairs so that the key space was reduced to the point of making practical brute force attacks against the key pair possible (Carnegie Mellon University 2017). A remote attacker could recover the private key that corresponds to a public key for keys that are less than 2048 bits long. The defect is in many cases found in hardware devices like TPMs and smart cards, and the solution to fix the defect is either to replace the device with one without the vulnerable RSA library implementation or install a firmware update.

In 2018, a cyber criminal group attacked the PIR Bank of Russia by using an outdated router as an attack vector (Amir 2018). The router was installed in a domestic branch of the bank. The attackers were able to transfer ca. 1 million USD to other accounts in other banks. The case was initially covered by the Russian newspaper Kommersant.

In 2018, researchers at Check Point Software identified two vulnerabilities (CVE-2018-5925 and CVE-2018-5924), named "Faxploit", in HP Inkjet network printers (Check Point Software 2018). The vulnerability was is the fax protocol, and therefore the affected device and software base is likely to be much larger, and the vulnerabilities have most likely existed for a long time. The researchers demonstrated how the vulnerabilities could be exploited to gain access to internal network by sending an image as a fax to the affected printer from the outside over public switched telephone. The affected printer with fax functionality, when attached to the telephone network, provides entry to the internal network and allows the traffic to bypass perimeter defences. The researchers suggest network segmentation, software updates and end point security as a defence against this kind of attacks.

Security issues can also be found in individual components. Multiple CPU design issues, the first of them named Spectre and Meltdown (Graz University of Technology

2018), were described in several disclosures during 2017 and 2018 showing that most widely used CPU architectures for mobile devices, desktops and servers have been vulnerable to data leaks for a long time through issues in how speculative execution has been designed and implemented. Some of the problems had existed for a couple of decades. There is variety in the outcome of each problem, but many of them allowed unauthorized access to sensitive data like passwords and breakout from virtual machines, potentially letting the attacker access all virtual machines in the host. Since basically all processors used in computer systems were affected, there was no easy immediate way out by replacing the affected hardware. As of summer 2018, the final solutions for all these vulnerabilities were still pending while new similar issues were emerging. One of them concerned Intel's Software Guard Extensions (SGX) where design issues in speculative execution could leak sensitive data (Wired 2018).

4.3.3 Hardware security in MediMaze

In the example case, hardware security could be an issue both for the corporation's information systems and its product line. The information systems can contain components that are later found to be vulnerable. If a network appliance in the office is vulnerable and it is connected to more than one network segments, there may be an opportunity for escalating a network attack through the appliance and provide access to private network, and in worst case, to business-critical infrastructure.

The hardware components in the medical devices may be vulnerable, requiring firmware or hardware updates.

4.4 Software security

4.4.1 Description

Software security refers to security issues that originate from defects in software code, making it behave in a way other than what the software designers and developers intended.

Defects in software code are a major source of data breaches and other incidents. The fundamental problem in software quality is the difficulty in removing all defects from the code during development or during its whole lifecycle.

Approaching the problem mathematically, having an average of 0,5 defects per 1000 lines of code (LOC) for an application in production use is a good achievement. For example, the Red Hat 7.1 operating system is estimated to have ca. 30 million LOC. With that number of lines and assuming quality code having 0,5 defects per 1000 LOC, approximately 15000 software defects are expected to exist in one system's operating system alone. A network of 100 such systems would have 1,5 million software defects before installing additional software like a hypervisor, container runtime software, guest operating systems, language middleware, application servers and business applications.

Some of these software defects are security problems, and with the high number of defects, chances of having security weaknesses in a system is likely to be high.

Avoiding software defects in software development is not in the scope of this thesis.

4.4.2 Software security issues in practice

Security defects can be reported and published as CVE (Common Vulnerabilities and Exposures) records, providing a source for statistical analysis and software security trends. CVE records contain known security defects of publicly released software packages only. In addition to that there are unknown vulnerabilities that have not been found so far, zero-day vulnerabilities (a vulnerability not known to those interested in fixing it but known to someone else with potential of exploiting it) and vulnerabilities in proprietary software, none of these tracked in CVE records.

According to SonicWall Cyber Threat Report 2018, more than 12500 new CVE records were created during 2017 (SonicWall 2018). Risk Based Security (RBS) reported in their Vulnerability QuickView that in 2017 there was almost 20 000 new vulnerability records in their vulnerability database (Risk Based Security 2018). These numbers were an all-time high. 39,5 percent of these vulnerabilities have an exploit available, while half of the vulnerabilities have a remote attack vector.

One CVE database is US National Vulnerability Database (NVD). The NVD website maintains statistics of vulnerabilities over their date of disclosure and their severity (NIST 2018). The statistics from 2001 to October 2018 are shown in Figure 4. The data shows a growing trend in amount of all disclosed vulnerabilities between 2001...2018. The last year in the figure is provisional and does not cover the last three months of the year.

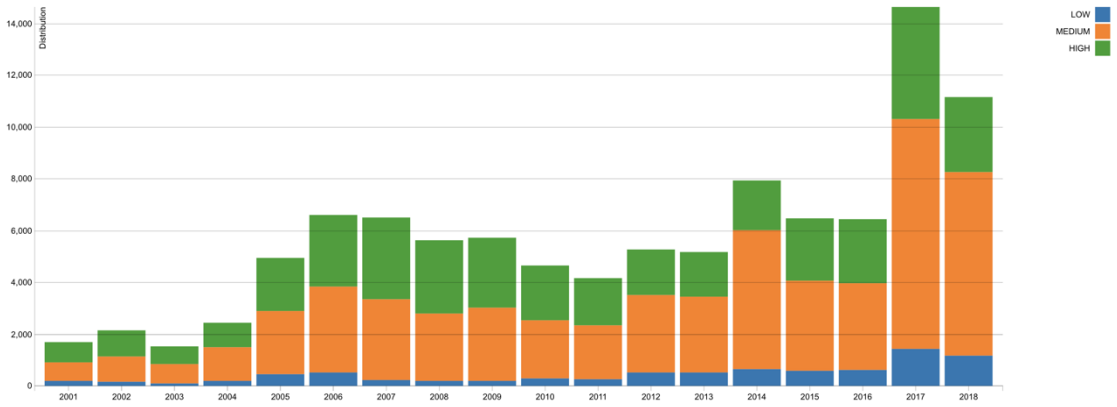

Figure 4. Statistics of vulnerabilities in NIST vulnerability database (NIST 2018).

The amount of vulnerabilities alone is not an issue but requires them to be severe enough to cause security problems. The Figure 4 also indicates that severity of the vulnerabilities is not improving over the period. This view is supported by Kuhn *et al.* (Kuhn, Raunak and Kacker 2017), who point out in their follow-up study using data in the NVD that 90 to 97 percent of all filed vulnerabilities over period of 2008...2016, a more limited time span than the NIST's statistics shown in the previous paragraph, were of medium or high-level issues. Severity of the vulnerabilities has not seen significant improvement. One observation in the same study that could explain part of the perpetual problems is that new IT tends to produce new IT security challenges while old challenges are slowly improving with better awareness and better tools (Kuhn, Raunak and Kacker 2017). The trend was observed when web-based services started to emerge in large scale, giving birth to web based vulnerabilities. One of the current problems is misconfigured cloud environments, discussed earlier in chapter 4.7.2, exposing data to leaks and breaches.

The pattern of new IT introducing new IT security problems appears to be repeating itself in Internet of Things (IoT) industry. One problem is the large scale and heterogeneity of network connected devices with limited resources and lacking security features (Zhang, et al. 2014). One notable result of this problem was the birth of Mirai botnet in 2016 (OVH 2016). The botnet consists of connected IoT devices, mostly IP cameras, digital video recorders and home routers. The purpose of the botnet is to perform DDoS attacks on selected targets. The Mirai malware compromises devices that are configured with default user credentials. This is possible because of either poor device management where the default credentials are left unchanged, or because of poor device security implementation where default user credentials cannot be changed. The original list of usernames and passwords the Mirai malware used contains just 62 entries. This was enough to harness a botnet large enough to produce the first DDoS attack that exceeded the bandwidth of 1 Tbps.

Kuhn *et al.* (Kuhn, Raunak and Kacker 2017) also note in another study that more than 2/3 of the problems are related to implementation while the rest are related to design and configuration. The implementation defects should be good candidates for removal by improving work practices such as doing better testing, doing better code reviews and using static code analysis tools.

As a conclusion, there seems to be little reason to believe that the problem of software defects causing security issues would be solved in the foreseeable future.

4.4.3 Software security in MediMaze

In the case of MediMaze, software defects can exist in all software used in the company and in all software used in the medical equipment. These defects can cause a wide scale of security issues in terms of integrity, confidentiality and availability, and cover all digitalized areas in the company. The software used in the business-critical infrastructure including all the medical devices in production use are subject to this threat. Software related issues in the medical equipment can endanger life, which happening in scale could endanger the existence of the company. Listing the likely defects and their outcomes is not practical to do here. Methods to mitigate security-related defects in software development are outside the scope of this thesis.

4.5 Cryptographic system and protocol security

4.5.1 Description

The class of cryptographic systems and protocols refers to use of secure cryptographic systems and protocols to protect data in rest, transit and use, as well as selection of relevant cryptographic systems for each situation, use of selected cryptographic systems in a secure way and security of cryptography system implementations.

Communication protocol and cryptographic implementations are complex software constructs and can cause security problems as any other software implementation. A protocol may fail in protecting confidentiality of information because of absence of security features or defects in design or implementation of them. A cryptographic algorithm may fail because of defects in the theoretical foundation or insufficient capability to resist brute force attacks, which may be a design choice to reach specific resource restrictions in an implementation.

A cryptographic system can have an insecure configuration like too weak randomness in key generation. Use of a cryptographic algorithm can lead to "technology lock-in" where change of cryptographic algorithm requires changing the keys, redistributing them and reworking all the already encrypted data. Weak cryptography is difficult to improve because of lacking agility in cryptographic methods and implementations.

Protocols and cryptographic systems can be successfully attacked by, for example, measuring their execution in some way, like measuring and comparing time or energy consumption of multiple runs and then concluding what happened in the system. This is referred as a side channel attack.

Another source of security issues is failure of cryptographic key lifecycle management for all keys from creation to usage, storage and destruction. Failure in key management can lead to leakage of keys, presence of keys with no owner in system and presence of keys with an owner outside of the organisation. All these failures can provide unintended access to data or systems. Keys can leak because of compro-

mised access control providing access to keys, exploit of a vulnerability providing access to keys, cleartext communication not keeping the key confidential, side channel attack, physical access to key storage, misuse of a cryptographic protocol, factors leading to human errors, and other reasons.

4.5.2 Cryptographic system and protocol issues in practice

An example vulnerability of a cryptographic protocol implementation is the ROBOT attack, or "Return of Bleichenbacher's Oracle Threat" (Böck, Somorovsky and Young 2017). This is a repetition of a chosen-ciphertext attack on RSA PKCS #1 v1.5 (first standard in Public Key Cryptography Standards by RSA Laboratories, version 1.5) encryption used in SSL, described by Daniel Bleichenbacher in 1998, and now found again in many TLS-RSA key exchange implementations. A protocol implementation may be vulnerable if static RSA encryption is used instead of forward secrecy where RSA is only used for signing. The issue is most likely a result of vendors not testing their protocol implementations against old known attacks (Böck, Somorovsky and Young 2017).

The vulnerability has many variations, and attacks has been demonstrated against implementations of all versions of SSL and TLS. Authors of the discovery demonstrated a practical attack of this vulnerability by successfully recovering the private key corresponding to Facebook.com's HTTPS certificate and signing a message using the key (4).

All TLS versions have countermeasures for this type of attack, but the increasing complexity of the countermeasures in each TLS version and difficulty to implement them right can undermine the goals. The variant of the vulnerability that made the Facebook private key compromise possible originated from a custom patch in the OpenSSL software. The ROBOT vulnerability affected a third of TOP-100 websites in Alexa Rank TOP-1000000, including Facebook.com and PayPal.com. In addition, many products from hardware and software vendors like F5, Citrix, Radware, Cisco were affected (4).

Because the attack operates on the private key, the attack has potential of becoming devastating when a static key exchange is used, instead of creating a new key for each session, and the traffic is not protected by some additional encryption protocol.

The attacker can passively record encrypted data communication over extended period of time, and after recovering the private key he can decrypt all the recorded data.

While the ROBOT is a protocol implementation issue, the KRACK, or "Key Reinstallation Attacks", is a protocol specification issue in IEEE 802.11i amendment covering WPA and WPA2 protocols (Vanhoef and Piessens 2017). The issue is in the 4-way handshake that is used for authentication and session key exchange. Similar issues are found also in PeerKey handshake, the group key handshake and the Fast BSS Transition handshake, all parts of the same specification. The vulnerability lets a malicious actor to reinstall an already-in-use session key by manipulating and replaying handshake messages. This can happen during an ongoing session. The key reinstallation vulnerability makes data confidentiality protocol (CCMP, TKIP or GCMP) vulnerable for attacks, which lets the attacker compromise confidentiality and integrity of the communication if it is not protected by an additional encryption protocol. The attack has a number of variants, and all the existing wireless LAN devices that use WPA2 are assumed to be affected, because the defected handshakes are mandatory parts of the protocol specification.

This case is interesting also in a sense that the vulnerability was in the part of the specification that was formally proven to be secure (Vanhoef and Piessens 2017, 14). The reason for this is the amount of ambiguity in the specification regarding key reinstallation. Therefore, the defect was not detected by the model used in the proof. Passing the formal proof probably made the community less interested in challenging the security of implementations, which allowed the vulnerabilities remain undetected for 14 years. The suggested countermeasure for all attack variants is to disallow reinstallation of the key that is already in use. This modification can be done to existing implementations without violating the specification.

How these particular protocol implementation and specification issues would affect the protocol usage? These cases question how secure and trusted wireless networking should be regarded even when the data in transit is encrypted. Wireless links are wide coverage broadcast links by their operating principle, lacking physical security. As shown by these two cases happening at the same time, even running a two-layer encryption scheme with two relatively well trusted protocols, WPA2 and TLS 1.2, was

vulnerable for their defects for more than a decade, covering the whole lifespan of the protocols at the respective date of discovery of these vulnerabilities.

The final example of security issues regarding cryptography, or lack of it, is a finding in the Gemalto Breach Level Index 2017, stating that only 3.1 percent of the 1756 breaches that leaked 2.6 billion records occurred for encrypted data (Gemalto 2017).

4.5.3 Cryptography and protocol security in MediMaze

In the example case, cryptography and protocol related security issues could start from a key management problem where a leaking key exposes stored or transmitted data to unauthorized access. A leaked key could also provide unauthorized access to corporate systems, causing confidential information to leak to the public or competitors and possible loss of integrity in the systems, data, configurations, source code repository and others. As a result, the medical devices could be provided with maliciously modified software updates, or updates could be denied. That could affect the devices' intended operation and eventually endanger human health and life.

Loss of private keys of the medical devices in production use would require reinstallation of the new public keys to the affected devices.

4.6 Interoperability

4.6.1 Description

Interoperability refers to artefacts, hardware or software, capable of interacting with each other to exchange information and then use that information. The interaction mechanism can be, for example, a standard communication protocol, a well-known encryption scheme, a language runtime or application data that is stored or transported in a certain format. These mechanisms are prone to changes over time as new versions of mechanisms are introduced in new protocol versions, operating systems and application versions during their normal evolution. Interoperability is therefore an important consideration in digital systems, particularly in business-critical infrastructures. Software updates may contain changes that can break interoperability, causing loss of availability, but they may also cause loss of confidentiality when a failure in a protection mechanism occurs.

Concerns regarding interoperability issues may lead to a decision to not upgrade something to a more recent and more secure or otherwise improved version but stay in the older version. This decision can be driven by desire to avoid the work and cost required in updating or upgrading the actual object and its dependencies. This causes creeping technical debt in systems which may eventually become security issues difficult to cure.

4.6.2 Interoperability issues in practice

An example of an interoperability problem was difficulties in creating a protocol version negotiation for TLS to be compatible with the existing mass of older implementations. Since protocol endpoints may not be using the same protocol version, there must be a way of agreeing which version is used. As explained by Nick Sullivan in Cloudfare's blog (Sullivan 2017), this negotiation phase has been broken in many SSL/TLS server implementations, and workarounds had to be implemented. One such workaround was "insecure downgrade", where a client downgrades its intended SSL/TLS version and then tries to reconnect the misbehaving server until a valid server response is received. After the discovery of the POODLE vulnerability in SSL3 protocol in 2014, malicious servers started to exploit it via the insecure downgrade mechanism by not responding to client connects until it proposed the vulnerable SSLv3 protocol. This became one example of how interoperability problems can escalate into security problems.

The Bleichenbacher attack on TLS protocols described in the previous section has also an interoperability viewpoint. While creating the TLS specification and finding methods to prevent successful Bleichenbacher attacks on the protocol, the same less secure RSA PKCS #1 v1.5 padding format (one of the two key ingredients in the ROBOT vulnerability) used in SSL was selected instead of the newer and more secure RSA OAEP (Böck, Somorovsky and Young 2017, 19). The reason for this was to maintain compatibility with older implementations. To mitigate the known security issues, countermeasures were designed to the protocol. These mitigations became larger and more complex over time in each TLS version, leading to bad implementations with ROBOT vulnerabilities.

As an example how an interoperability problem between two applications may lead to propagating security problems is a situation where business is dependent on running an application that is not supported and therefore is not receiving updates or patches. Its dependency, e.g. a language runtime, is still receiving updates, and therefore it may eventually cease to be backwards compatible with the business-critical application. Updating the dependency would break the application. If there are other applications with the same dependency as the problematic application, they may evolve to versions that are not compatible with the common dependency that was left without updates. Eventually, all these applications have fallen outside of updates, and known vulnerabilities may be left without patches. Now the known vulnerabilities start to pile up.

The origin of this propagating security issue was the business-critical application with no support. However, the application and its dependencies did not have to be vulnerable to create the security issue. This example demonstrates why an interoperability issue should be regarded as an emerging security issue, and why simple systems with minimum functionality, minimum role, minimum dependencies and therefore minimum risk of escalating interoperability problems within a system should be preferred over complex systems with multiple roles and complex network of dependencies. In addition, loss of support should lead to phasing out the application.

4.6.3 Interoperability security in MediMaze

In the example case, interoperability problems can occur when systems in the critical infrastructure are upgraded to their newer versions, or new software or hardware are adopted to use, but clients of these software and hardware are not yet fully compatible. This could lead to various issues like availability of the service or functional problems in the medical devices while, for example, trying to interact with the affected services. Loss of interoperability could lead to problems in installing updates to medical devices. The medical devices may also receive an update that is not fully compatible with the device. All software and hardware in the business-critical infrastructure will eventually reach their end of life, and a possibly complicated upgrade process may lead to decision to not upgrade until a later time.

4.7 Configuration security

4.7.1 Description

In this thesis, configuration security refers to security issues relating to composition of something and including the corresponding settings that prescribes the behaviour of the application, device or host. The definition includes access control configuration. Examples of configurations are applications made of software components, a system made of applications and user accounts, and a network of systems, each with settings for each application, user account and the operating system.

This is a wide topic with many opportunities for exploitable weaknesses. The weaknesses can result e.g. from an accidental mistake, uninformed action or a deliberate action on access control systems, privilege settings of user accounts, settings of software and hardware, installation of online network devices, installation of malicious software in systems. Many of these can be sources of security issues without any known vulnerabilities in software code.

In practical implementations, this class must be divided into subclasses according to individual needs to get attention to areas with special security needs. For example, there could be software configuration, server configuration, network configuration, user account configuration and access control configuration subclasses.

4.7.2 Configuration security issues in practice

According to Gemalto Breach Level Index Report 2017, incidents involving accidental loss of data increased from under 250 million records in 2016 to 2.6 billion records in 2017 (Gemalto 2017). The causes of these breaches come "largely from poor security measures protecting external assets like websites and backup systems as well as misconfigured systems like publicly readable and writable AWS S3 buckets".

User accounts are a popular attack vector. Typical issues are easy-to-decrypt passwords and password reuse over many accounts in different services, but a user account security issue may also result from a hard-coded user credentials in devices and applications, effectively backdooring the application or the device. These ac-

counts are not created by a legitimate user but by someone during software development or by a malicious actor after compromising the system to ensure persistent access.

Users with too high privileges can be a security problem by allowing too much access for attackers. According to the Avecto Microsoft Vulnerabilities Report (Avecto 2017) concerning the year 2017 for Microsoft operating systems, Office products, internet browsers and servers, removing local administrator privileges from users would have mitigated 80 % of critical vulnerabilities reported in Microsoft products in 2017, 79 % of the critical vulnerabilities in all Windows operating systems, 94 % of the vulnerabilities in Microsoft Edge and Internet Explorer, 74 % of the critical vulnerabilities in Windows Server, 60 % of vulnerabilities in Microsoft Office and 80 % of all critical remote code execution vulnerabilities. The report emphasizes the need for enforcing the principle of least privilege in all user accounts and suggests use of endpoint privilege management to balance between security and usability (Avecto 2017, 8).

The "Faxploit" case described earlier in hardware security (Check Point Software 2018) also has a configuration security viewpoint. Having a fax or any networked device located in the junction point of two or more networks may provide an opportunity for pivoting, i.e. using a vulnerable host as a stepping stone for moving to other hosts in a different network. These kinds of vulnerabilities can be mitigated by segmenting a network to group devices and then defining rules to manage traffic flows between the groups in a way that prevents this kind of movement.

In 2018, web infrastructure of British Airways was penetrated in a targeted attack and the Modernizr JavaScript library file was modified by adding 22 lines of JavaScript code to create a cross-site scripting attack that leaked customer and credit card data of 380 000 people during the 15 days when the customers were doing online payments using their web browsers and mobile applications (Klijnsma 2018). This attack was an example of how an unauthorised change in existing configuration can compromise security and have major consequences. One way to add defence against these kinds of attacks (in addition to keeping the primary controls effective) is to implement file integrity checking for files that do not change often and are likely targets in attacks.

4.7.3 Configuration security in MediMaze

In the example case of MediMaze, configuration issues could occur when a device containing sensitive data for internal use is accidentally configured to be accessible from a public network without access control despite of its classification in the policy to be for internal use only. The issue would effectively change the configuration of the network by exposing a device to hacking attempts and possibly to a leak of its sensitive data, and after a successful penetration, provides the attacker persistant access to the host in the internal network via a backdoor. This host may become a pivoting point to attack hosts in the business-critical infrastructure. With ability to control the host, the attacker may be able to listen to the host's environment and provide the data to the attacker over an extended period of time. Unconventional channels like DNS and ICMP tunnels may be used for data exfiltration.

The software produced for the medical devices may use software packages that are not what they were assumed to be, making the devices behave in ways not intended. This could expose the devices for attacks and leak sensitive information from them.

A configuration issue may occur when an employee installs an unauthorised device into the internal network, or installs unauthorised applications to a workstation. These installations may open attack vectors that were not there without the installations.

A user account related issue may occur when an employee leaves the company, but the user account remains active. The employee or the company may leak the user account's access credentials and provide access to an outsider.

4.8 Supply chain security

4.8.1 Description

Few organizations can operate successfully in isolation. They need partners and third-party suppliers to run their business. In a digital context it brings digital interaction to the business interaction, which in turn brings many opportunities to break security.

Supply chain security is defined in this thesis to cover everything an organisation receives or uses as external resources that have capability of causing security issues. Examples of such resources are vendor-provided software, third party software packages, hardware installations, external services such as cloud-based storages and services, and business partners.

The key characteristic of this attack method is that it attacks trust relationship between two parties, where a security weakness in a trustee's system provides an easier way to perform an intrusion into a trustor's system or its data than attacking the target directly.

4.8.2 Supply chain security issues in practice

According to the study by Ponemon Institute in 2018, 59 % of organizations in the USA and UK have experienced a data breach caused by one of their third-party suppliers (Ponemon Institute LLC 2018). One method of performing a supply chain attack is to attack software repositories and their packages to ease access to other hosts and their data. An example of this method was a Python software package "SSH Decorator" that, after having several clean versions in its history, changed its behaviour and started to upload users' private SSH keys, passwords, usernames and other related data to a website (Bleeping Computer 2018). This case is a practical reminder of why auditing all versions of all software packages is important.

Another method is to set up a malicious mirror software repository and provide its clients with old software packages with known vulnerabilities despite of the availability of newer versions in the main repository. Another method is to upload a software package with a typosquatting name (name that closely and intentionally resembles a well-known name) with malicious code in it, in hopes of having it accidentally downloaded into software projects. Yet another method is to buy a software project and then modify it for malicious intents.

Another case was the NotPetya malware in 2017, where a mass of systems was attacked by first compromising the M.E.Doc accounting software update server that was vulnerable, and then changing a file in the software package to create a backdoor in the victims' systems (ENISA 2018, 56). The outbreak was most notable in

Ukraine and its near region where over 80 percent of businesses were using the software.

Supply chain attack can also be carried out by exploiting third party partner's access to the target or by using something obtained from a victim that can be used for malicious intents. As an example, a highly advanced case of creating and deploying a cyber weapon and using supply chain attacks for successful completion was the operation named "Olympic games" or "Stuxnet" (Langner 2011). The operation compromised programmable logic controllers (PLC) inside an airgapped network to sabotage uranium enrichment centrifuges in Natanz nuclear facility in Iran. The initial penetration to the facility was carried out by means of a worm that was initially delivered to five contract companies that had physical access to the nuclear facility. This was the first supply chain attack in the operation. Then the worm propagated from a Windows system to another using portable mass storage devices as vehicles over airgaps. When the worm eventually found a Windows system running Siemens Simatic Step 7 automation system software (assumed to be responsible for running the centrifuges), it installed a driver file (effectively a rootkit) in two types of PLCs that were connected to the Simatic system. The drivers made the centrifuges misbehave beyond human control and break themselves apart. To remain stealthy in all the Windows systems along the route to the target system, the malware code was signed using authentic but stolen software signing certificates (Kushner 2013). This was the second supply chain attack in the operation.

In addition to demonstrating threats in supply chains, this is a practical example on why airgapping a network is not a sufficient method to secure digital infrastructures since it can only prevent a limited range of attacks. The threat may still creep to the protected side of the airgap from actions such as software installations and software updates from physical media, installation of hardware and use of portable mass storages, none of them affected by having an airgap around the network.

4.8.3 Supply chain security in MediMaze

In the example case, a supply chain attack can target MediMaze or its clients.

The initial attack can occur in anywhere where software, hardware or services are produced for MediMaze. Third-party software packages used in software projects

may be altered in the vendor's end, and pulling the compromised packages to MediMaze's software projects can make the medical devices misbehave. The compromised devices may join a botnet and carry out actions the attacker commands, like participate in denial of service attacks or cryptocurrency mining. These activities could lead to various side effects such as reduced performance and stability in the device's primary functions, endangering human health or life.

The software used in the business-critical infrastructure may be compromised at the vendor's systems and open backdoors for attackers.

Third-party partners may use compromised laptops to access MediMaze's internal network and infect vulnerable hosts in the network with malware.

A supply chain attack from MediMaze's clients' viewpoint could occur when a compromised device management software is downloaded from MediMaze's software repository to client systems, or when a compromised medical device scans for vulnerable hosts in the hospital network and installs malware when a target host is found.

4.9 Human error

4.9.1 Description

Human error is defined in this thesis to mean human actions within a complex system (e.g. a network of computer systems) resulting in negative outcomes that were not intended by anyone. This definition excludes cases where a malicious actor is involved in the situation. Cases where someone is victimising an employee are covered later in security awareness and capability.

Human actions contain human errors which may mix up with technical failures. Examining security issues in digital environments from a technical perspective only may not reveal the root causes of issues that are observed but only one side or an interphase of them.

Human error could be seen as an outcome of problems in human information processing between inputs and outputs. This view could then explain human error as a cause for a failure. Another view on human error is to understand it as an outcome

of systemic factors deeper in the system. In research of safety critical industries, failures resulting from erroneous human actions while operating complex systems are found to be outcomes of a wide range of cognitive, collaborative and technological problems (Woods, et al. 2010, 3). Human error is in this view an outcome, not a cause for failures. Human error is an endpoint in scale for human performance and the other endpoint is human success. The problem to tackle is complexity, not humans (13).

When human error is understood as erroneous human actions while operating an apparently faultless system, the obvious solution to the "human problem" is to reduce humans' role in the system by replacing human work with automation or placing new rules, controls and other restrictions on human work.

However, these solutions will not fix the real problems and therefore will not prevent others from repeating the same error. Automating mechanic human work may help in some cases, but it may also transfer human role towards the blunt end of the system. Research have shown that organisations successful in preserving safety are characterised by ability to continuously understand vulnerabilities, anticipate and plan for changing and unexpected events and future surprises instead of just avoiding risks and preventing errors (11-13). This is one way how people adapt to complexity.

4.9.2 Active error vs. latent error

Woods *et.al.* describe how complex systems have a *"sharp end"* and a *"blunt end".* The sharp end is where frontline operations happen. Erroneous human actions and the resulting failure occur in the same time and space. Because of these actions' active role in the outcome, these errors are called *active human errors* (Woods, et al. 2010, 51).

The blunt end consists of activities such as design, manufacturing, training and management. Errors in these activities occur in other time and space than the resulting failures and require a triggering event to activate their effect and consequences. These errors are called *latent human errors* (51).

Reasons behind active human errors causing incidents are likely in multiple latent human errors laid earlier in the blunt end of the complex system. When they are triggered and the error passes through imperfections in defences, an active error in the sharp is likely to occur (51).

4.9.3 Teamwork viewpoint

As stated in chapter 4.9.1, one source of human errors is problems in collaboration. We can translate this to mean problems in utilizing potentials of teamwork. For correct and effective decisions, a team needs to maintain team level situational awareness by sharing information. This is particularly necessary when working in environments where the amount of data can be overwhelmingly high, and where bad decisions can lead to dire consequences (Åhman and Gustafsberg 2017, 34). Bad decisions originate from human, environmental and systemic factors (34). Communication, common strategies, common practices and common goals can mitigate this. However, team members do not need to have the same situational awareness to work successfully. When there are outages in information sharing, a shared mental model of the situation can help the team in maintaining situational awareness in dynamic situations (35).

A decision-making error can result from an observational error or from bad motivation (132). To eliminate decision making errors, Åhman and Gustafsberg describe a process to identify these errors, with a note that a key to preventing decision making errors is in making and testing them within teams having people with diverse way of thinking, preferably to the degree of making the situation "slightly uncomfortable" for the participants (133).

4.9.4 Human error related issues in practice

As discussed earlier in configuration security, 2.6 billion records were leaked in breaches during 2017 (Gemalto 2017). According to the report, nearly 2 billion, or 77 percent, of them were a result of "accidental losses", which was more than 3 times more than the second largest source, leaks caused by malicious outsiders (6). Incidents in that year were characterised by "poor security practices" in form of misconfiguration of cloud storages, web servers and backup systems. The percentage is in

line with other industries such as airline industry, air traffic control and nuclear power, where approximately 70 to 90 percent of incidents are related to human errors.

One case that contributed to the leak of records is the Equifax breach in 2017 that leaked personal information of 145.5 million American, British and Canadian citizens. The breach exploited a critical command injection vulnerability in Apache Struts (CVE-2017-5638) used in an online dispute portal that was unpatched at the time of the incident. The patch had been available for 2 to 4 months at the time of the breach (Bergel 2017). We take this case for closer examination since United States Government Accountability Office (GAO) investigated the breach and released a report in September 2018 describing the events and the steps taken by the organisation to recover from the incident (United States Government Accountability Office 2018). The report does not give any recommendations, nor does it examine the probable root causes in depth. Therefore, the conclusions presented here are interpretations of the report.

The attack started two days after US-CERT had disclosed the vulnerability when adversaries scanned public systems for the vulnerability. The attackers found the Equifax's dispute portal to be vulnerable and tested effectiveness of their exploit by running test commands on the server. These events went undetected.

The vulnerability existed because a critical security patch was not applied to the system running the dispute portal. Equifax had a practice of informing system administrators about vulnerabilities through internal mailing lists. The recipient data was out of date, and the information about the vulnerability did not reach the administrators responsible for the attacked server, while the other servers were updated successfully (15).

The report does not state if this was the primary or secondary channel for vulnerability information sharing inside Equifax, or how the mailing list was maintained. As a primary channel, as it appears to have been considering the outcome and the mitigations on the issue later, it is prone to human errors when maintained manually. This characteristic may have contributed to the success of the breach.

Interpreting the report, the opportunity for the error was provided by latent human errors such as decision to use the rather clumsy and error-prone method for sharing information that is critical to maintain corporate security, instead of opting to use dedicated tools that would provide system administrations the security critical information directly.

Later, the attackers gained access to the dispute portal again. The adversaries were able to escalate their attack beyond the 3 databases associated with the dispute portal after finding one of them containing unencrypted login credentials. Missing data governance allowed the attackers access this data without restrictions. This allowed the adversaries to escalate their attack to 48 other databases in the network that were not related to the dispute portal (18).

Interpreting the report, storing credentials unencrypted was a decision making error in the Equifax administration. The same applies to absence of network segmentation and data governance, which helped with escalating the attack. In this thesis, these errors would be classified as configuration security and security awareness failures with probable causes in earlier decision making in administration, a latent human error.

There was a "device", likely an IDS sensor, looking for malicious traffic in the network. The device did not work at the time of the breach because its digital certificate had expired 10 months earlier, making the device unable to inspect the encrypted traffic during the 76 days the attack was active (14). A simple active human error in forgetting to update a certificate was enough to eliminate the capability of detecting ongoing attacks, which contributed to the success of the breach. Updating the certificate enabled the device again, and that was when the attack was detected and stopped.

Interpreting the report, absence of an effective process or other mechanism to ensure that security critical certificates are updated in time, instead of relying that practitioners at the sharp end remember to carry it out, was likely due to latent human errors in the administration, providing an opportunity for the active error to happen.

The report identified four main problem areas that contributed to the breach: problems in detecting malicious activities, problems in identifying vulnerabilities, weaknesses in segmentation and missing data governance. Interpreting the report and reflecting the issues to the human error description, all four problem areas were a result of systemic failures in Equifax organisation's security practices, originating from the decisions that made the organisation work the way that was not effective in resisting active human errors from escalating into security problems. This, in turn, allowed adversaries to access the systems and exfiltrate large amount of sensitive data. The issues were later fixed in terms of better tools, improved configuration and revised policies (17).

Four conclusions can be made from these interpretations. First, the root problems behind the success of this breach were not technical but rather related to decision making long before the incident.

Secondly, the failure of the operational staff to update the affected server, the "human error", was not the root cause behind the breach. It was the systemic problems in Equifax's way of managing and maintaining its digital security. As a result, the operational staff at the sharp end had limits in their capability of working successfully. Remarkably small deviations in the sharp end from the intended course of actions resulted in the major breach.

Thirdly, there was no one root cause to the incident, but separate causes in two parties, Apache project and Equifax.

Finally, events had multiple aligned steps from root causes to the breach through imperfect defences. It was not just a single event in time and space, the actual breach. There were many opportunities available to avoid it from happening, and they all obviously failed.

4.9.5 Human error in MediMaze

In the example case, human errors are likely to occur in all human work, including manual labour and decision making. More devastating errors in the frontline operations are likely to happen when working *on* security controls, in comparison to work-

ing *under* them. Examples of security critical activities are updating or planning access control settings, making system changes using superuser privileges and working on the production environment.

4.10 Security awareness and capability

4.10.1 Description

In this thesis security awareness and capability is defined as understanding requirements for a secure way of operating, and capability of putting the knowledge into effective action. This class also contains situations where someone is victimized by a threat actor and lured into unsafe actions.

This class deals with the “blunt end” as described in chapter 4.9.2. The human role in the blunt end is to appreciate the practice at the sharp end to anticipate paths to failure and support their robustness and resiliency (Woods, et al. 2010, 12). The organisation’s role is then to provide the necessary resources to enable frontline operations to be successful in their efforts.

The threats in this class are then related to inadequate or missing resources to keep the business-critical infrastructure secure. In this case the threat actor can be the management when refusing to provide enough resources for security related work, and possibly maintaining a diluted security culture. These factors can make the organisation systemically unsecure.

4.10.2 Security awareness and capability issues in practice

Keeping things safe is becoming increasingly difficult not only because of the sophistication and diversity of attacks, but also because of new technology along with digitalisation and networked objects that makes businesses vulnerable to new threats and new enemies. More challenges come from narrow time scale from the vulnerability disclosure to an incident, and from shortage of skilled staff to work on security.

The short time the threat actors can create or change their attacks in has been demonstrated in a number of cases. One of them was in 2018 where a highly critical vulnerability (CVE-2018-7602) in Drupal CMS was exploited widely using an exploit that was published seven hours after the vulnerability was disclosed. This was too

fast for many site administrators to react (Bleeping Computer-2 2018). For comparison, in tCell's Security Report for In-Production Web Applications (Q2/2018) it was found that the mean time from discovery to patch for vulnerabilities in web applications was 38 days (tCell 2018).

As stated earlier, cyber security is not just a technical issue. There is likely a human element in the issues in some way. One such element is human misbehaviour under social engineering where an attacker entices an insider human into doing an action or giving information to damage security without understanding the situation and its consequences. Examples of such actions include installing malware by opening a compelling file provided by an attacker, paying fake bills, letting malicious people inside the company premises for a fabricated reason, or by giving the attacker sensitive information like login credentials.

A study by Taimur Bakhshi (Bakhshi 2017) presents an example of reaching a target and carelessness of employees under a social engineering attack. In the study, two different experimental spear phishing attacks were carried out in a corporation to see how many responses the attacks produce. In the first test, an email with a link to an external web form asking for information about the corporation was sent to a target group. Approximately 46 percent of the receivers opened the email and clicked the link. Half of them completed the form. In the second test, 15 USB sticks with a label "confidential" in them and the TrustedSec social engineering toolkit installed were left in the corporate premises for employees to recover. Of the 15 devices, 60 percent were eventually plugged to a workstation and a file in the stick opened.

This experiment, albeit having a narrow focus in relation to all security awareness issues, shows how employees can be a significantly weak link in organisation's security. Hacking humans can provide malicious actors an initial access with less resistance than hacking its systems. This is most likely the reason why social engineering is an increasingly popular attack method. This can be seen in Webroot's threat trends report 2017, stating that in average 1.4 million new phishing sites were created each month. Phishing was used in estimated 90 percent of all security incidents, making it the top cause for breaches and other security incidents (Webroot 2017).

Targeting humans shows also in results of a survey by Osterman Research (Osterman Research 2018), stating that between March 2017 and March 2018 approximately

one organization out of four reported that their systems were infected by malware in a phishing attack and data had been leaked, including sensitive and confidential information such as user credentials.

Shortage of skilled cyber security staff is another factor in capability of maintaining good enough security. A questionnaire study made by Enterprise Strategy Group and Information Systems Security Association to security and IT professionals (Oltsik 2017) showed that the majority of organizations have experienced one or several security incidents over the past two years. The shortage of cyber security staff is one contributing factor, while lack of training, cyber security process bloat, managers' lack of security knowledge and treating cyber security with low priority were among the other challenges. Ninety-one percent of survey respondents believe that most organizations are either extremely vulnerable or somewhat vulnerable to a damaging cyber attack. The responders were mostly from the USA (85 % of all responses).

4.10.3 Security awareness in MediMaze

In the example case, the difficulty in maintaining adequate state in security is likely to remain a persistent balancing issue between having security and keeping the cost at a reasonable level. Management may restrict costs too much and have too few employees in security work. The employees may not get enough learning opportunities and not enough equipment to have their work done as they wish. Employees may not understand the tactics used by malicious actors and fall victim to targeted scams. Also, management or team members may adhere to having a blame culture where accidental errors and failures lead to personal blame and consequences. The working culture may not be security aware, in that daily decision making contain little or no security related viewpoint, daily working habits contain a lot of security risks and corner-cutting is commonplace with no-one challenging it.

4.11 Conclusion of the threat taxonomy

These nine classes chosen from the given sources are subjects for threat outlook in this framework. As was seen in the descriptions and the example cases, security issues may have characteristics from more than one class.

The threat taxonomy described here answers research question Q1.

5 Framework for securing business-critical infrastructures

5.1 Definitions

Having the purpose of the framework described in chapter 2, and the threat taxonomy described in chapter 4, next targets and their content, goals, are defined to glue the framework together.

5.1.1 Goals

In this framework, cyber defence is driven by high level and far-reaching goals. The goals describe the desired end state of security and the direction for improvements over time. All goals together cover all nine classes in the framework. How far each goal should reach is a decision of the implementer depending on the desired outcomes.

As an example, a goal stating "there are no known vulnerabilities in systems" is practically impossible to fulfil, however, it can be approached indefinitely. Each step in approaching this goal may reduce the time window between disclosure of vulnerabilities to patching them in systems. This can be regarded as a desired improvement in security.

5.1.2 Targets

Targets are collections of goals. Each target collects goals that affect an area of activity in security.

This thesis identifies need for three targets:

- *structured environment* for understanding the environment and how it affects the defensive activities,
- *structured work* for making plans, decisions and other work on defensive actions that secure business activities,
- *structured systems* for implementing computing resources to assist the structured work to be successful.

There are five requirements regarding targets based on the goal of the framework to reduce security gaps.

- All targets should contain goals for all classes with the emphasis that reflects the identified need and sophistication for security outcomes.
- All defences should relate to at least one goal.
- Goals must not conflict with each other.
- There must be defences for each goal.
- Only defences that do not conflict with the goals should be accepted in targets.

The first rationale for these requirements is to have enough defensive items to protect the critical infrastructure with the least amount of gaps.

The second rationale is to avoid complexity and excess in security and its implementation, which could lead to difficulty in understanding how the implemented security works, how it should be organised and if the security as a whole works in the intended way. A practical example of this issue is adding technical security solutions with high privileges in systems, which also adds high-risk attack surface.

When goals are fulfilled by concrete defences, such as security controls and various actions, they become members of the corresponding target. Thus, targets are collections of goals and their implementing defences.

The definitions given next for all three targets can be used as a starting point for the refinement work that eventually leads to complete definition of the targets.

5.1.3 Structured environment

In his thesis, the term "structured environment" contains far-reaching goals related to having a complete and up-to-date understanding of the cyber-physical environment and the related phenomena in which the people work using their tools and other resources. As a result, the team gains capability of planning and reacting to threats.

By definition, the environment contains only elements and events that are outside of the employees' and their organisation's direct control, yet they may have effects on both. This includes inside threats.

The starting point for goals is described in the following list:

- Structured environment is information about prevailing and future cyber-physical threats.
- Structured environment is information about the organisation’s past security outcomes with successes and failures.
- Structured information is current and describes the environment as it is.
- Structured information is relevant and concrete and can be used for decision making.

The information base can be collected from provided resources like literature, written documentation, threat reports and news, training, internal research, research publications, from other people e.g. peer employees and contact networks, or from technical sources like security information and event management systems.

The structured environment improves over time, providing new useful information while phasing out old information on the cyber-physical environment which then improves structured work and structured systems towards their goals.

This term is an opposite to unstructured environment where there is little or no understanding of current cyber-physical environment, leading to a reduced level on situational awareness, which contributes to late or missing decision making, inefficient and obsolete security practices, irrelevant implementations, inefficient teamwork and added workload. These lead to unexpected surprises and events that may have negative consequences because of a missing plan on how to react. The resulting work is reactive and corrective instead of proactive and preventive, possibly done under pressure and stress that is a discouraging and prone to errors.

5.1.4 Structured work

In this thesis, the term "structured work" contains far-reaching goals of organising human work to be security aware, effective and streamlined in all tasks and activities so that cognitive load and use of resources is minimised. The structured work relates

to turning information and knowledge into improvements in everything that relates to human work for the nine classes. The starting point for goals is described in the following list:

- Structured work is aware of all assets at all times and the necessary level of defence. All systems and data have protection against realistic threats using a proper amount of resources regarding the assets' value.
- Structured work refines information about its cyber-physical environment for decision making.
- Structured work aims to maintaining a complete understanding on how security works in the networks and systems.
- Structured work aims to make the people at the sharp end capable of acting effectively and timely for both routine and unexpected situations using using prepared plans, tools, principles, procedures and situational information that minimize cognitive load.
- Structured work improves teamwork and team level information sharing.
- Roles and responsibilities inside the team are clear with enough authority and resources given to each role to carry out effective actions when necessary. The skill requirements for each role are clear and opportunities for improvement are available. The roles are set so that there are no gaps between them leaving necessary capabilities outside of the defined roles.
- Structured work has no unnecessary, overlapping or redundant tasks, and no necessary work is left undone in time.
- Structured work learns from failures by encouraging open information sharing and using the information to improve the working practices and tools. Failures are regarded as opportunities to understand problems and their root causes, learn something new, and then improve security by preventing repetition of failures.
- Practices, procedures and systems are documented, and the documentation is always up-to-date, providing adequate support for all tasks and minimizing human errors. The documentation is free of entropy.
- Structured work improves over time, and each change is a step closer to the goals presented here. One target for improvement is the list of goals for structured environment, structured work and structured systems.

Structured work is an opposite to unstructured work that is characterised by redundant or overlapping tasks and other inefficiencies that produce waste, work not done in time, reliance to guesswork and opinions instead of correct information. Work is reactive instead of proactive because of missing information and weak awareness of the current situation and thread landscape. The inefficiencies lead to rush and other

deficiencies prone to produce human errors and accidental security issues. Tools are inappropriate or unsuitable for the tasks, wasting time. Blame culture obstructs information sharing and learning from failures. Frontline operations are dependent on management and external bodies in daily decision making. Roles are missing or unclear, creating skill gaps. All this leads to inferior security with ineffective or obsolete practices and implementations that tends to worsen over time, leading to ever increasing chance of having security incidents.

5.1.5 Structured systems

In this thesis, term "structured systems" contains far-reaching goals on characteristics of systems and tools on how they are designed, implemented, configured, secured and documented so that they support the human work in the best way. The definition regards systems, software, security controls, hardware, their settings and their documentation. The finished set of goals covers all nine classes in the taxonomy.

The presented list of goals is a starting point and can be extended and refined to reflect needs of each implementation. All these goals should be technology agnostic to ensure their relevancy as system and security implementations evolve over time.

- There are no unpatched known vulnerabilities, default passwords or similar predictable free rides into systems.
- Security tolerates a hostile system in an arbitrary network segment capable of carrying out attacks and listening to network traffic according to commands given by a malicious actor.
- Systems detect unauthorised connections, unauthorised installations, unauthorised processes, unauthorised hosts, unauthorised user accounts and unauthorised data exfiltration, and act once these are detected.
- Structured systems allow access only to known subjects with least amount of privileges needed to accomplish the subject's tasks.
- Structured systems aim at a minimum role for each host, minimum amount of code and minimum amount of internal and external dependencies, however, may seek for diversity in their implementation to avoid systematic weaknesses in scale.
- Structured systems are designed, implemented and minimized to provide the intended outcomes, and no else. Systems detect, resist and report activities they are not intended to do or experience.

- The systems and software are allowed to interact with other entities only when it contributes to desired outcomes. All other access and interaction are disabled and prevented at all levels.
- System organization, operation and security is designed and implemented to be clear for those who work with each system.
- Systems provide relevant information to improve situational awareness that supports the work to improve security.
- There are no single points of security failures.
- Systems and their configuration are always well known. Unauthorised changes to configuration are detected and reversed.
- Systems and services have enough fault tolerance that reflects their importance. This includes ability to evade denial of service attacks that introduce excessive amount of traffic to the target.
- Structured systems can be recovered after a loss within reasonable predefined time to a predefined point.
- The software help employees and other stakeholders to reach their goals with minimal mechanical work, cognitive load and cost. The software have a correct set of functions with optimised usability in each of them. No unnecessary functions exist. User interface functionality is self-sufficient.
- Structured systems have data lifecycle management in place so that the systems contain, store and handle sensitive data they need and only when it is necessary. Data and data storages that have reached their end of life are disposed with no data remnants left behind.
- Structured systems improve over time, and each change is a step closer to the goals presented here.

Structured systems are an opposite term to unstructured systems where systems are not well understood because of their complexity, systems and their security is not well understood because new artefacts are added and old ones removed to meet a specific need at the time. Documentation is poor and makes changes difficult to plan and problems difficult to fix. Accumulation of ad hoc and custom modifications to software, systems and networks over time without proper change management reduces comprehension on their operation, maintainability and security, and administration of the systems become exceedingly difficult, adding cognitive workload and opening opportunities for accidental failures. Software may not have the latest patches installed for various reasons, and software versions may be old and exceedingly difficult to upgrade because of, for example, growing interoperability issues. Recovering such a system after a loss is difficult. Data lifecycle is not managed, with

data left in memory, mass storages and backups and is exposed to leaks and unauthorised access. All these problems tend to worsen over time with a result of having ever increasing chance of having security incidents. Employees find the systems discouraging to work on, possibly leading to staffing problems that feeds the problems further.

5.2 Putting the framework together

There are now nine classes and three targets. Relationship between classes and targets is also clear: classes contain threats while targets are containers for goal-driven defences. The finished framework is shown in Figure 5.

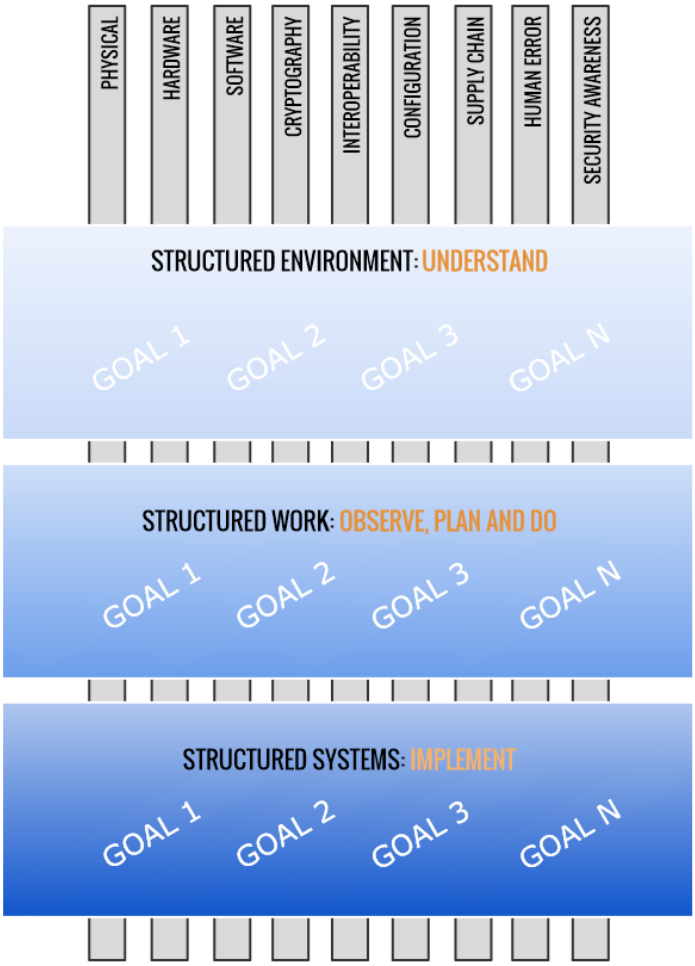

Figure 5. The framework with three targets containing goals that cover all nine classes. The class names are truncated.

When implementing the framework, goals for each target-class pair need to be created to the extent that reflects the desired level of security. This is an opportunity to form a desired emphasis for security as well. To fulfil the goals with actual defences, we can use for example technical controls, methodical descriptions related to human work, information, instructions, actions, plans, documentation, action calendars, skills, training, capabilities and knowledge. This principle is shown in Figure 6.

This thesis will not provide a list of concrete defences since they are implementation dependent.

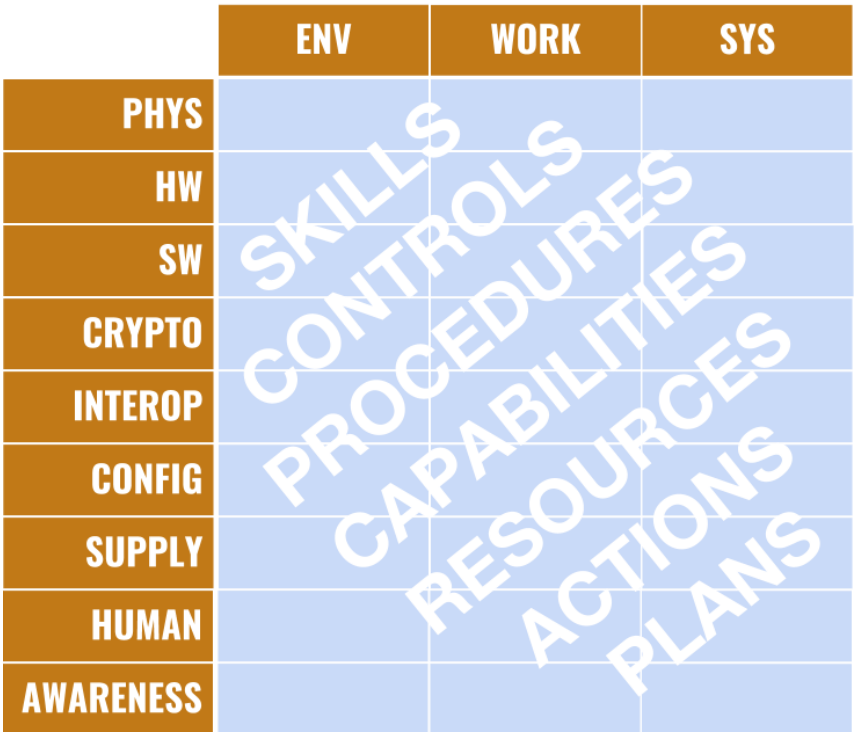

Figure 6. Framework filled with various kinds of defences for each class-target pair. The class and target names are truncated.

5.3 Refining target boundaries

For proper use of the language of this framework, the target boundaries need to be clear. When deciding the correct target for an activity, the answer may become clear by asking "what is *directly* affected or improved as an outcome of this activity? Is it

human work, concrete systems or information about the cyber-physical environment?".

Controls may contain characteristics of more than one target and class. For example, a control that mandates "collecting log data and reviewing it periodically" consists of installation and configuration of log collecting software (targeting structured systems) and then reviewing the collected data (targeting structured work). These can be classified into *software security* in structured systems and *security awareness and capability* targeting structured work and possibly to *configuration security* of structured systems.

As another example, when the work is done to make an improvement to a system, the activity is classified as structured systems because the direct outcome is an improvement in the systems and not in the work itself. Reading data from systems for supporting decision making is classified as structured work since the outcome is directly related to supporting work and not to make improvements to the systems. The information *may* be used to improve systems, however, that is to be decided later.

When the work is done using the tools and information sources to gather or refine information that improves understanding of the environment, the activity is classified as structured environment, because the direct result is information and not improvements in work or systems. When tools are installed, configured and maintained to collect such information, the activity is classified as structured systems because only the systems are directly improved. When working methods are defined to improve information collection from the environment, the activity is classified as structured work since the direct outcome is improvement in work.

Employees are regarded as something there is control over (employees are under direct command and can be removed from the organisation if necessary), so an activity of, for example, logging employees' actions is not a target of structured environment but a target of structured work. Malicious insiders are under some control, but humans outside of the organisation are not under direct control, so activity of collecting data from them for threat information is a target of structured environment. This distinction is needed because controls against insider and outsider threats are different and are likely to affect both structured work and structured systems in a different way.

The finalised framework construction described in this section answers research question Q2.

5.4 Using the framework to reduce systemic security gaps

The fulfil the purpose of the framework, implementation should aim for maintaining a continuous effort covering all parts of the organisation that is directly related to the protected infrastructure, instead of trying to implement it as a single effort or on a per-asset basis. More focused implementation may be necessary when the infrastructure is experiencing significant change.

1. Initiate with dividing each class to subclasses so that their content and scale are manageable to avoid missing areas that are likely to need attention. No classes or targets are omitted.
2. Start with structured environment. Collect threat and other security information from various sources that appear to be relevant and related to all classes in the framework.
3. Continue with structured work. Based on the collected information, plan and document work in terms of plans, skills, capabilities, resources, procedures, instructions and others that are needed to defend successfully against the identified threats. Do this for every class in the taxonomy.
4. Finish with structured systems. Tools are selected, implemented, configured and secured to support the structured work. Use sources of technical information that helps in each specific task.
5. Repeating steps 2…4 frequently during normal work in fine-grained cycles covering all classes for all assets over time.

Repeating the three targets over the nine classes in a continuous and fine-grained fashion is the method of this framework to obtain the known characteristics of organisations where practitioners are successful in operating their system safely under their threat environment (as discussed in chapter 4.9): understanding vulnerabilities, anticipating possible paths to failures in changing environment, and planning and adapting for various situations.

A more specific example of using of the framework is presented later in chapter 6.6.

6 Benchmarking the framework against other frameworks and standards

6.1 Benchmarking overview

The framework is now ready, and we can assess it. This part of the thesis tests the framework for its coverage against all available controls in the two reference frameworks and their control lists.

To see how well all the controls presented in the reference frameworks cover areas presented in our framework, the controls are benchmarked to find if this framework can cover at least as many or more areas in security than the other two frameworks.

The hypothesis is that this framework covers at least the same number of areas as the other two. If the areas of this framework are not covered by the other frameworks, or coverage is weak, it can be assumed that this framework can provide new areas to look at while finding controls to protect the digital business-critical infrastructure. Finding meaningful controls indicates usefulness of this framework in that particular implementation. Since the other two frameworks and standards are well established in the industry, it can be assumed that a comparable or better coverage would suggest usefulness of this framework.

6.2 Experiences from the benchmarking process

The benchmarking proved to be an important part of this thesis since it provided input for improvements for the framework like a practical testing would have provided. This part of the work forced to review and rethink all the definitions more carefully than what was initially done. All class and target definitions required improvement and refinement to have a strict and clear enough meaning and function for them. Prior to the refinements the concepts were too ambiguous, leaving too much room for opinion and too much chance for variation of the benchmarking results. After each refinement the benchmarking process was started over again with the better class and target definitions. This reduced uncertainty of the results and reduced chances of having different results for different benchmarking runs.

6.3 Benchmarking the controls of ISO 27001 Annex A

First the hypothesis was tested by benchmarking this framework against ISO 27001 Annex A to see how well its controls cover classes and targets the framework. In the examination each of the 146 controls are taken and decided which class-target pair the control fits the best.

The results of the benchmarking are seen in Figure 7. The full list of control mappings from Annex A to our framework is shown in Appendix 4.

	ENV	WORK	SYS
PHYS	0	12	15
HW	0	0	0
SW	0	3	2
CRYPTO	0	3	4
INTEROP	0	0	0
CONFIG	1	22	21
SUPPLY	0	6	0
HUMAN	0	2	0
AWARENESS	1	49	5

Figure 7. Benchmarking results of ISO27001 Annex A against the framework. Names for classes and targets are truncated.

Results show that controls are mostly related to defining security related work, less so securing systems and almost none with understanding the environment. There are several directives to create policies and procedures in the control list, and these belong mostly to work to build security awareness and work on building secure configuration, so these controls gravitate to structured work regarding security awareness

and capability and configuration security. A major part of these controls deal with "information" or "assets", so they were classified as security awareness or configuration depending on each control.

6.4 Benchmarking the CIS controls

Next the hypothesis is tested by benchmarking the framework against CIS controls to see how well its controls cover classes and targets in our framework.

As in the previous benchmarking, all 194 controls presented by CIS controls are taken and put into our framework in the corresponding class-target pairs. Then the coverage is examined if there are gaps that need to be addressed with other controls. It should be noted that all CIS controls are not likely implemented in real world implementations because of its extensiveness. Therefore, representativeness of the results has their limits, but the results are assumed to be good enough for conclusions.

The benchmarking results are shown in Figure 8. The full list of control mappings from CIS controls to our framework is shown in Appendix 5.

	ENV	WORK	SYS
PHYS	0	0	1
HW	0	0	0
SW	1	14	6
CRYPTO	0	2	9
INTEROP	0	0	0
CONFIG	0	27	78
SUPPLY	0	0	0
HUMAN	0	2	0
AWARENESS	1	42	16

Figure 8. Benchmarking results of CIS controls against the framework. Names for classes and targets are truncated.

Results show that the overall coverage of CIS controls is similar to ISO 27001 Annex A. However, the CIS controls have more emphasis on configuring systems, where work relating to configuration security and configuring systems has most controls. Work to improve security awareness and implementing the related work also has a plenty of controls, with software security in terms of work and implementations in systems also having coverage.

The weak areas are similar to ISO 27001 Annex A, with structured environment almost uncovered, with the same set of classes either uncovered or having weak coverage.

6.5 Conclusions of the benchmarking results

The finding of the results is that our framework is only partially populated by both reference control lists. The situation would not change significantly if both control lists were combined because they have similar overall coverage on our framework.

The second finding is zero or near-zero coverage for many areas in our taxonomy. These two reference frameworks have no or weak coverage for hardware security, interoperability security and human error. Coverage for supply chain security is missing in CIS controls and limited in ISO 27001 to cover structured work. Physical security is covered in ISO 27001 while coverage in CIS controls is weak.

Both references have most coverage for security awareness and capability. This class is quite wide with many controls. Our framework has rather coarse granularity here, and a benchmarking with configuration security divided into several subclasses could reveal additional results.

In terms of targets, understanding the environment of each area in security appears to be weak in both reference frameworks, with only two controls over the nine areas in both references. This is a surprising result and may be explained by the intended audience of these references. It is also possible that the references are intentionally less explicit about this in their control lists, and hence, will not provide results during the benchmarking.

Structured work is generally well covered by both references with more emphasis by ISO 27001 that CIS controls. Structured systems have more controls in CIS controls than in ISO 27001.

We can also see how the references' controls are emphasised. Both benchmarking results show mostly similar overall emphasis in our framework with a few notable exceptions. First, the CIS controls are more system configuration oriented than ISO 27001, with 39 % of the controls populating the configuration security in structured systems against 14 % for ISO 27001. Second, the weight on structured work on security awareness has more weight in ISO 27001 (34 %) than in CIS controls (21 %).

We can make the following final conclusions from the results. Our framework appears to have more emphasis on understanding the environment for all nine classes than the reference control lists. Since the effective defensive actions are based on understanding the threat environment, this finding is rather unexpected. In addition, our framework proposes coverage for hardware security, interoperability security, supply chain security and human error, which all have weak or no coverage in the reference frameworks. This is also a surprising result since the literature study revealed great importance of human error and supply chain issues is breaches happening in the field.

Both ISO 27001 and NIST framework recommend collecting controls from other sources that these. This benchmarking results supports the recommendation when business-critical infrastructure is being protected.

The true value of our framework would emerge when having the missing areas filled with meaningful defensive actions. We give an example of this in the next section.

This examination answers research question Q3.

6.6 An example of utilising the framework

We next fill the gaps of missing or weak coverage in the ISO 27001 Annex A and CIS controls with additional defences. For improvement, there may be needing to understand high level security related tasks and their organisation, and there may be needing to find appropriate technical controls to secure the infrastructure.

As an example, we see in benchmarking results of our framework against the two references that they have weak coverage in controlling *human error*. As discussed in the literature review, this is a common source of security issues and therefore needs attention. Instead of going through all classes, we demonstrate the framework usage just for the class of human error.

As explained earlier in chapter 5.2, our framework approaches the problem in three steps: first understanding the environmental matters, then arranging the work to build controls and practices to defend against the threat and finally setting up tools that support the work. All these targets are goal oriented as defined in chapter 5.1.1, and all the defensive methods can be connected to one or more of the goals.

6.6.1 Understand the environment

We start adapting the approach suggested by our framework by doing the following practical steps to understand the matters we have no direct control on:

- learn what human error is, how and why these errors happen, and how they manifest themselves in reality,
- learn what kind of human errors could happen and how, and what kind of human errors have happened in our organisation or elsewhere in the past,
- learn what areas in the protected infrastructure might be subject for human errors,
- learn what are or have been their root causes,
- study relevant literature on how effects of human error could be prevented or mitigated, then applying the information to the case.

After completion of these items we create a base to build the actual defences and controls that mitigate the identified issues caused by human error. The outcome may contain a generic list of possible issues that human errors may cause, or more specific list that reflects the recent experiences in the organisation.

6.6.2 Plan and improve work

Now when we have a basic understanding of the problem, we can move on to work on the necessary changes in how our team or organisation works. This phase could have the following steps:

- study how security critical work is done in the organisation and where the possible deviations towards failures could occur,
- study where a single erroneous human action (or a missing action) could result in a security event,
- study what kind of unexpected situations are likely to arise sooner or later,
- study what changes are needed to the current way of working to mitigate the identified potential problems,
- learn about common pitfalls and misconceptions on applying the planned changes to the working practices,
- plan how to migrate to the new way of working in an effective way without causing vulnerabilities during the change,
- write new or improved instructions, actions and other supporting documentation to reduce human errors, and make the documentation available and easier to approach to the affected members of the organisation,
- introduce new working methods to daily work and show how the affected tasks are done from now on, what improvements they provide and explain why it is important,
- plan for follow-up items to see how the changes are affecting our organisation.

Depending on the outcome, the controls on mitigating human error may be done by introducing changes similar to the items in the following list:

- simplification and streamlining of procedures to have less steps, less mechanical work and less temptation to optimize and go with a less secure way,
- encourage a practice of avoiding security critical decision making and actions in a hurry or under significant stress,
- preparing for unexpected situations with plans where quick and effective security critical actions may be necessary,
- where found relevant, creation and use of written guidelines, check lists and action calendars for actions and procedures to ensure that security critical tasks, regardless of how routine they are, are done and can be done without loading the human cognitive capabilities or trusting the practitioner's memory excessively, and have the work finished as intended,
- where found relevant, introduction of a practice of doing security critical actions that are characterised by being security critical and difficult to revert with no defined procedure available, as a teamwork by explaining own perception of the situation and intentions to another team member for agreement before committing to actions,

- where found relevant, have a practice of cross-checking security critical actions made by a team member to ensure that the task really ended as intended,
- conducting security reviews on work outcomes to identify security problems,
- finding ways in terms of working habits, procedures, instructions, knowledge, skill, improved tools, communication and willingness of learning from failures to avoid repetition of the same security critical errors by different people,
- encouraging to have a practice of challenging all noticed security corner cutting and other misbehaviour, accidental or intentional, before they become a new normal,
- introduction of practice of accepting input and questions from all team members for a basis of security related discussion and decisions,
- encouraging team members to speak aloud of all matters that may have security implications,
- ensuring that there is no blame culture in the organisation that could discourage open discussion about security problems or failures.

When improvements in work are planned, we can proceed to plan for improvements in systems that support our work.

6.6.3 Plan and improve systems and tools

Improvement in systems may include introduction of new tools or modifications to existing tools and systems to make them more focused and relevant to the work. The improvements may relate to provision of current, accurate and easy to understand situational information. Systems and tools can be improved in their user interfaces by removing unnecessary functions to reduce complexity and mechanical work for better usability to all the functions that are relevant to the work. This could reduce the cognitive load and remove sources of unnecessary active human errors.

There may be need to phase out old software or hardware that are no longer needed or that have caused problems in the past. There is likely a need to follow-up how the changes are taking effect by making changes for producing follow-up data that can be later evaluated.

6.6.4 Other sources of controls

Depending on the problem and the class under examination, there may also be need for other controls and supporting information found in various sources. In addition to the already discussed sources in the benchmarking process, controls can be found in documents, such as NIST SP 800-53, CIS Benchmarks, Finnish Katakri National Security Auditing Criteria, and Finnish VAHTI (Government Information and Cyber Security Management Board) instructions.

6.6.5 Conclusion of the example case

This example shows how utilizing the way of applying all three target controls to a single threat class. Running the method continuously in fine-grained steps with constant improvements we establish a method to reduce the chances of having systemic security gaps in the organisation's business-critical digital infrastructure.

This example examination answers research question Q4.

7 Discussion and future work

This thesis was an initial experiment of constructing a framework that could help teams and other flat organisations with understanding the threats of their business-critical digital infrastructures and then direct their actions towards defending against the threats. The goal is to have less dark corners or gaps in security and have improved view on variety of threats on protected assets, which can provide a better foundation for effective risk mitigation work.

The framework was constructed to identify the most important areas where security may fail and how to continue from there. The classes were formed using input from a brainstorming session, ENISA threat taxonomy and Lockheed Martin's cyber kill chain. The resulting taxonomy of nine classes is assumed to be the minimum taxonomy for securing business-critical infrastructures.

The targets were formed in a brainstorming session where high level controls were studied. The three targets presented in this thesis are collections of goals for security outcomes. The end result is a structured framework having a clear three-layer structure made of targets, goals and classes. Structuring should reduce security gaps and reduce systemic problems in security implementation.

One interesting topic for further study is the use of agile methodology for continuous implementation of this framework. This approach would embed the security tasks into daily production work as sprints, and have the actions continuously implemented in the digital infrastructure much like results from software development sprints are continuously integrated into the main code base. This approach would help keeping the security implementation relevant against the changing threat landscape and resist accumulation of security debt. The agile methodology may be easier to implement in the context of this framework than in context of, for example, ISO 27000 because of its simplicity and structured design, both in attempt to make the framework suitable for teams and flat organisations.

To see how this framework compares in its coverage to existing widely adopted frameworks, and if this framework can present any additional coverage, this framework was benchmarked against ISO 27000 and CIS controls. The results indicate that our framework has more guidance to understand the threat

environment over all threat classes. Regarding security related work, this framework has coverage in interoperability, hardware security and human error that are almost completely uncovered by ISO 27001 Annex A and CIS controls. In addition, our framework suggests more explicit coverage for physical security and supply chain security that are almost completely uncovered by CIS controls. Finally, this framework is more explicit in suggesting controls for hardware security, interoperability, human error and supply chain security, which are almost completely uncovered by both ISO 27001 Annex A and CIS controls despite how important role these issues are in breaches and leaks of sensitive data in the field.

The benchmarking results and the known security issues found in the literature review hint that the ISO 27001 and CIS controls could benefit from having more explicit coverage for the aforementioned areas. More explicit coverage in understanding the cyber-physical environment, their threats and then doing the protective work with support from appropriate tools could reduce chances of having systemic gaps in these areas that are known to be important sources of threat. These areas are covered in our framework, indicating extended coverage and reduction of dark corners in security over the references. This, in conjunction with constructing the framework, is the result of the work.

References

Åhman, Helena, and Harri Gustafsberg. 2017. *Tilannetaju - päätä paremmin.* Helsinki: Alma Talent Pro.

Amir, Waqas. 2018. *Hackers attack Russian bank to steal $1m using an outdated router.* Accessed 24.7.2018. Retrieved from https://www.hackread.com/hackers-attack-russian-bank-to-steal-1m-using-an-outdated-router/.

Avecto. 2017. *2017 Microsoft Vulnerabilities Report.* Accessed 13.8.2018. Retrieved from https://www.avecto.com/resources/reports/microsoft-vulnerabilities-report-2017.

Bakhshi, Taimur. 2017. Social engineering: revisiting end-user awareness and susceptibility to classic attack vectors. *Emerging Technologies (ICET).* IEEE. 1-6.

Bergel, Hal. 2017. Equifax and the Latest Round of Identity Theft Roulette. *Computer* 50: 72-76.

Bleeping Computer. 2018. *Backdoored Python Library Caught Stealing SSH Credentials.* Accessed 23.5.2018. Retrieved from https://www.bleepingcomputer.com/news/security/backdoored-python-library-caught-stealing-ssh-credentials/.

Bleeping Computer-2. 2018. *Hackers Don't Give Site Owners Time to Patch, Start Exploiting New Drupal Flaw Within Hours.* Accessed 8.5.2018. Retrieved from https://www.bleepingcomputer.com/news/security/hackers-dont-give-site-owners-time-to-patch-start-exploiting-new-drupal-flaw-within-hours/.

Böck, Hanno, Juraj Somorovsky, and Craig Young. 2017. Return Of Bleichenbacher's Oracle Threat (ROBOT). *iacr.org.* Accessed 13.6.2018. Retrieved from https://eprint.iacr.org/2017/1189.

Carnegie Mellon University. 2017. *Infineon RSA library does not properly generate RSA key pairs.* Accessed 31.7.2018. Retrieved from https://www.kb.cert.org/vuls/id/307015.

Check Point Software. 2018. *Faxploit: Breaking the Unthinkable.* Accessed 14.8.2018. Retrieved from https://blog.checkpoint.com/2018/08/12/faxploit-hp-printer-fax-exploit/.

ENISA. 2018. *ENISA Threat Landscape Report 2017, ENISA report.* ENISA.

ENISA-2. 2016. ENISA threat taxonomy 2016 - a tool for structuring threat information. *Enisa.europa.eu.* Accessed 21.5.2018. Retrieved from https://www.enisa.europa.eu/topics/threat-risk-management/threats-and-trends/enisa-threat-landscape/etl2015/enisa-threat-taxonomy-a-tool-for-structuring-threat-information.

Gemalto. 2017. The Year of Internal Threats and Accidental Data Breaches 2017. *Breachlevelindex.com.* Accessed 16.4.2018. Retrieved from http://www.breachlevelindex.com/assets/Breach-Level-Index-Report-2017-Gemalto.pdf.

Graz University of Technology. 2018. *Meltdown and Spectre, Vulnerabilities in modern computers leak passwords and sensitive data.* Accessed 25.5.2018. Retrieved from https://meltdownattack.com/.

ISO/IEC. 2009. *ISO 27000:2009 Information Security Management System.* International Organization for Standardization.

Kananen, Jorma. 2017. *Kehittämistutkimus interventiotutkimuksen muotona. Opas opinnäytetyön tai pro gradun kirjoittajalle.* Jyväskylä: Jyväskylä University of Applied Sciences.

Kasanen, Eero, Kari Lukka, and Arto Siitonen. 1993. "The constructive approach in management accounting research." American Accounting Association.

Klijnsma, Yonathan. 2018. *Inside the Magecart Breach of British Airways: How 22 Lines of Code Claimed 380,000 Victims.* Accessed 12.9.2018. Retrieved from https://www.riskiq.com/blog/labs/magecart-british-airways-breach/.

Kuhn, Richard D, MS Raunak, and Raghu Kacker. 2017. An Analysis of Vulnerability Trends, 2008-2016. *Software Quality, Reliability and Security Companion (QRS-C).* IEEE. 587-588.

Kuhn, Rick, Mohammad Raunak , and Raghu Kacker. 2017. It Doesn't Have to Be Like This: Cybersecurity Vulnerability Trends. IEEE. 66-70.

Kushner, David. 2013. The real story of stuxnet. *IEEE Spectrum.* IEEE.

Langner, Ralph. 2011. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy.* IEEE.

Laukka, Lasse. 2015. *Information Security Management System Implementation for a CERT. Master's thesis.* Tampere: Tampere University of Technology.

Lockheed Martin. 2015. Gaining the Advantage - Applying Cyber Kill Chain Methodology to Network Defense. *lockheedmartin.com.* Accessed 31.7.2018. Retrieved from https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/Gaining_the_Advantage_Cyber_Kill_Chain.pdf.

NIST. 2018. *CVSS Severity Distribution Over Time.* Accessed 14.10.2018. Retrieved from https://nvd.nist.gov/general/visualizations/vulnerability-visualizations/cvss-severity-distribution-over-time#CVSSSeverityOverTime.

NIST-2. 2018. Framework for Improving Critical Infrastructure Cybersecurity. *Doi.org.* Accessed 10.6.2018. Retrieved from https://doi.org/10.6028/NIST.CSWP.04162018.

Ojasalo, Katri, Teemu Moilanen, ja Jarmo Ritalahti. 2014. *Kehittämistyön menetelmät - Uudenlaista osaamista liiketoimintaan.* Helsinki: Sanoma Pro Oy.

Oltsik, Jon. 2017. The Life and Times of Cybersecurity Professionals. *Issa.org.* Accessed 14.6.2018. Retrieved from https://c.ymcdn.com/sites/www.issa.org/resource/resmgr/surveyes/ESG-ISSA-Research-Report-Lif.pdf.

Osterman Research. 2018. Best Practices for Protecting Against Phishing, Ransomware and Email Fraud. *Ostermanresearch.com.* Accessed 24.5.2018. Retrieved from https://www.ostermanresearch.com/home/white-papers/.

OVH. 2016. *The DDoS that didn't break the camel's VAC*.* Accessed 8.5.2018. Retrieved from https://www.ovh.com/world/news/articles/a2367.the-ddos-that-didnt-break-the-camels-vac.

Ponemon Institute LLC. 2018. Data Risk in the Third-Party Ecosystem: 3rd Annual Study by Ponemon Institute. *Opus.com.* Accessed 11.11.2018. Retrieved from https://www.opus.com/ponemon/.

Risk Based Security. 2018. *Vulnerability Quick View 2017.* Risk Based Security.

Shostack, Adam. 2014. *Threat modeling: Designing for security.* John Wiley & Sons.

SonicWall. 2018. 2018 SonicWall Cyber Threat Report. *sonicwall.com.* Accessed 11.4.2018. Retrieved from https://cdn.sonicwall.com/sonicwall.com/media/pdfs/resources/2018-snwl-cyber-threat-report.pdf.

Sullivan, Nick. 2017. *Why TLS 1.3 isn't in browsers yet.* Accessed 25.5.2018. Retrieved from https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/.

tCell. 2018. Security Report for In-Production Web Applications. *tcell.io.* Accessed 23.8.2018. Retrieved from https://info.tcell.io/in-production-application-security-report-2018.

United States Government Accountability Office. 2018. Actions Taken by Equifax and Federal Agencies in Response to the 2017 Breach. *gao.gov.* Accessed 14.9.2018. Retrieved from https://www.gao.gov/assets/700/694158.pdf.

Vanhoef, Mathy, and Frank Piessens. 2017. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. *Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS).* ACM.

Webroot. 2017. *Webroot Threat report 2017.* Webroot Inc.

Webroot. 2018. *Webroot Threat Report 2018.* Webroot Inc.

Wired. 2018. *Spectre-Like Flaw Undermines Intel Processors' Most Secure Element.* Accessed 14.8.2018. Retrieved from https://www.wired.com/story/foreshadow-intel-secure-enclave-vulnerability/.

Woods, David, Sidney Dekker, Richard Cook, Leila Johannesen, and Nadine Sarter. 2010. *Behind human error.* Second. Farnham: Ashgate Publishing Ltd.

Zhang, Zhi-Kai, Michael Cho, Chia-Wei Wang, Chia-Wei Hsu, Chong-Kuan Chen, and Shiuhpyng Shien. 2014. IoT security: ongoing challenges and research opportunities. *Service-Oriented Computing and Applications (SOCA).* IEEE. 230-234.

Appendices

Appendix 1. Process of the first brainstorming session

The process of the first brainstorming session was carried out to identify and classify potential security issues a business-critical infrastructure could face. The taxonomy starts to form.

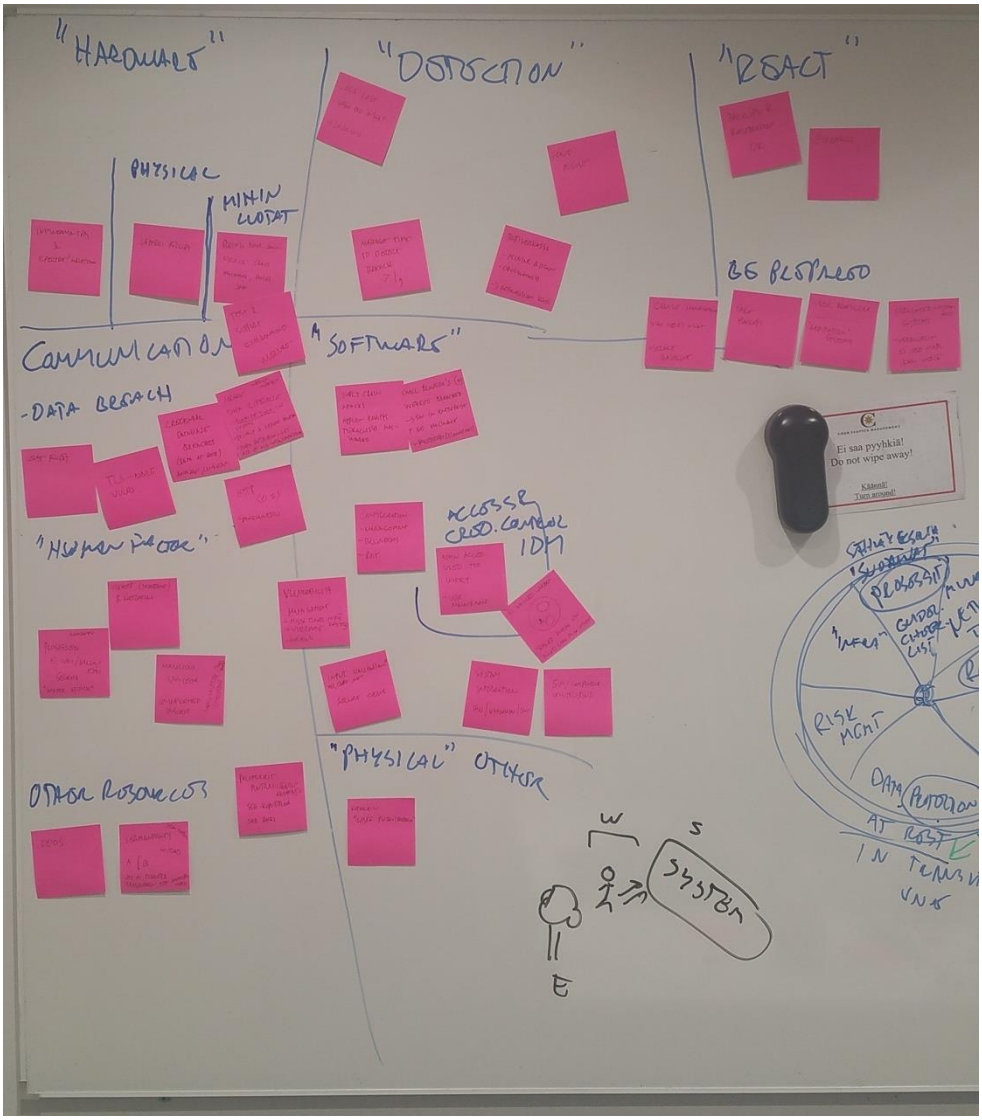

Figure 9. Process in the first brainstorming session to identify sources of threats.

Appendix 2. Result of the second brainstorming session

The second brainstorming session was an attempt to form a classification for the identified security issues. The resulting taxonomy has eight classes. The taxonomy is still not final since the other sources were not yet used at this stage.

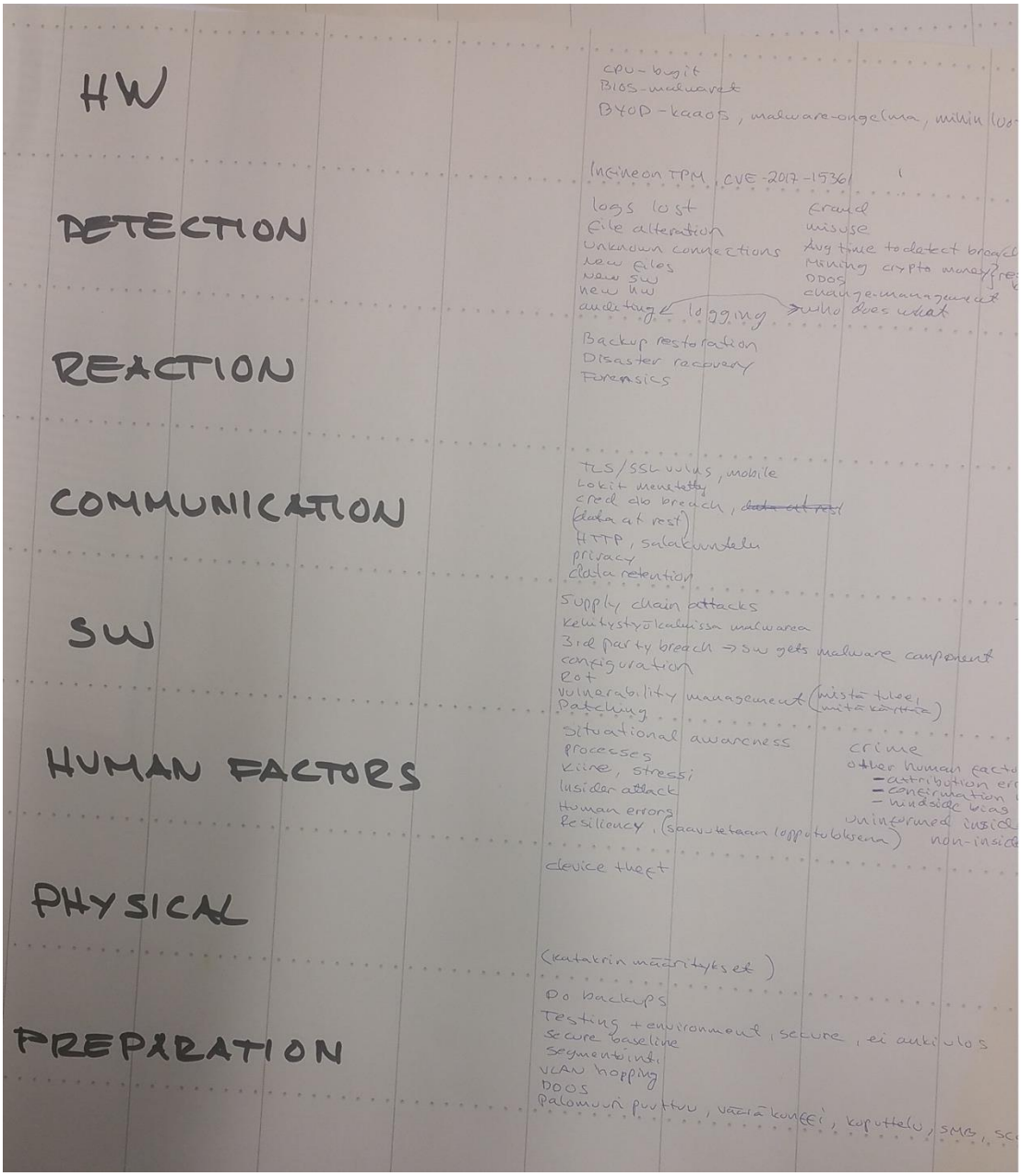

Figure 10. Result of the second brainstorming session to form a threat taxonomy for the framework.

Appendix 3. Targets emerge during the second brainstorming session

The second outcome of the second brainstorming session with three targets. The targets (structured environment, structured work and structured systems) are emerging from organisation of high-level controls. This set of targets was eventually the final version with no obvious needs to either extend or truncate it. However, their content changed considerably.

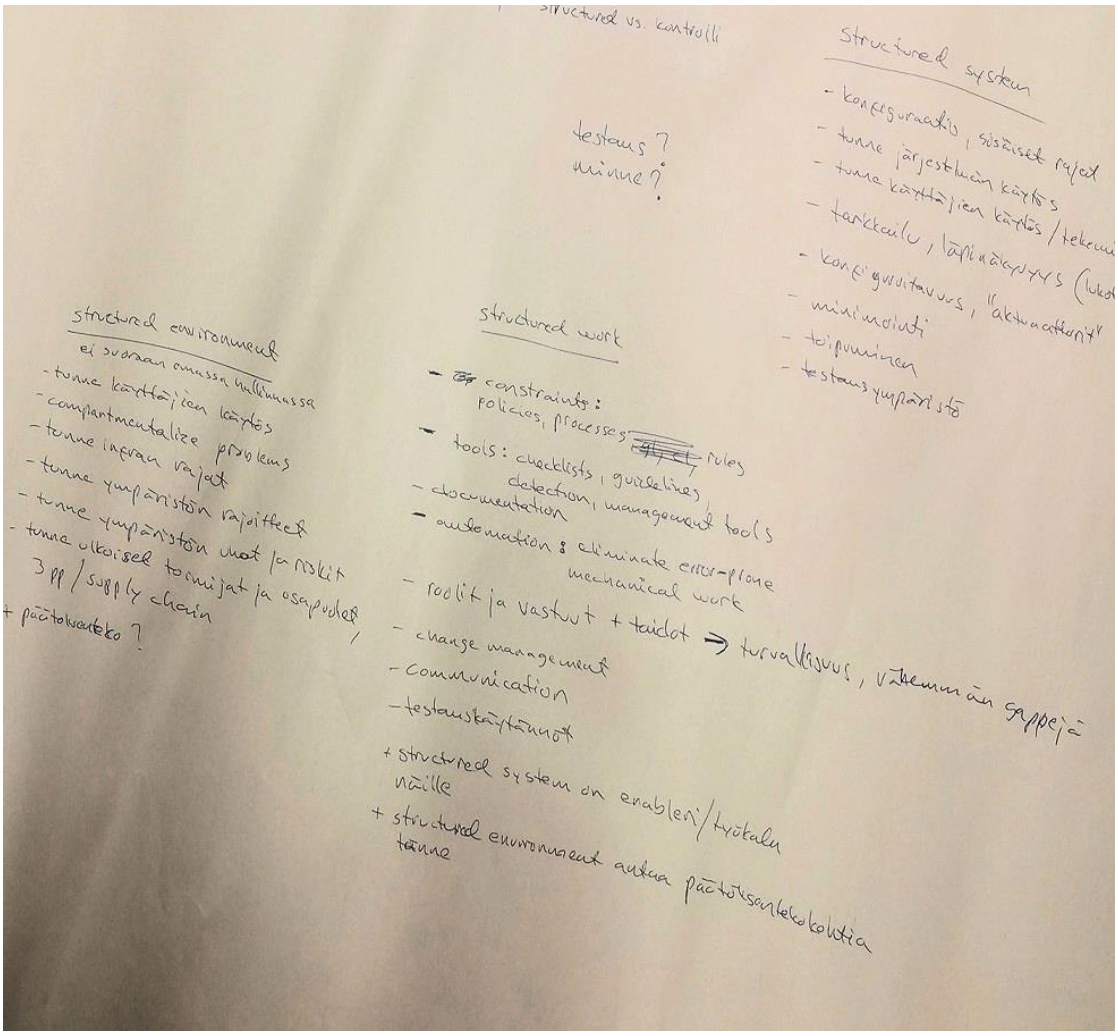

Figure 11. Outcome of the third brainstorming session to find classification for dealing with threats.

Appendix 4. Benchmarking of controls in ISO 27001 Annex A

This table contains the mapping of each control defined in ISO 27000 Annex A against our framework, 146 mappings in total, 31 of them found in more than one class. Class names have been truncated.

Table 1. Mappings from ISO 27001 Annex A to the framework presented in this thesis.

Control ID	**Target**	**Class**
A5.1.1	Structured work	Security awareness
A5.1.2	Structured work	Security awareness
A6.1.1	Structured work	Security awareness
A6.1.2	Structured work	Security awareness
A6.1.2	Structured work	Human error
A6.1.3	Structured work	Security awareness
A6.1.4	Structured environment	Security awareness
A6.1.5	Structured work	Security awareness
A6.2.1	Structured work	Security awareness
A6.2.1	Structured systems	Configuration security
A6.2.2	Structured work	Security awareness
A6.2.2	Structured systems	Configuration security
A7.1.1	Structured work	Security awareness
A7.1.2	Structured work	Security awareness
A7.2.1	Structured work	Security awareness
A7.2.2	Structured work	Security awareness
A7.3.1	Structured work	Security awareness
A8.1.1	Structured work	Security awareness
A8.1.2	Structured work	Security awareness
A8.1.3	Structured work	Security awareness
A8.1.3	Structured work	Security awareness

A8.1.4	Structured work	Physical security
A8.1.4	Structured systems	Configuration security
A8.2.1	Structured work	Security awareness
A8.2.2	Structured work	Security awareness
A8.2.2	Structured systems	Security awareness
A8.2.3	Structured work	Security awareness
A8.2.3	Structured systems	Security awareness
A8.3.1	Structured work	Physical security
A8.3.2	Structured systems	Physical security
A8.3.3	Structured systems	Physical security
A9.1.1	Structured work	Configuration security
A9.1.2	Structured work	Configuration security
A9.1.2	Structured systems	Configuration security
A9.2.1	Structured systems	Configuration security
A9.2.1	Structured work	Security awareness
A9.2.2	Structured work	Configuration security
A9.2.3	Structured systems	Configuration security
A9.2.3	Structured work	Security awareness
A9.2.4	Structured work	Configuration security
A9.2.5	Structured work	Configuration security
A9.2.6	Structured work	Configuration security
A9.3.1	Structured work	Configuration security
A9.4.1	Structured systems	Configuration security
A9.4.2	Structured work	Configuration security
A9.4.3	Structured systems	Configuration security
A9.4.4	Structured systems	Configuration security
A9.4.4	Structured work	Configuration security
A9.4.5	Structured systems	Configuration security
A10.1.1	Structured work	Cryptography security
A10.1.1	Structured systems	Cryptography security

A10.1.2	Structured work	Cryptography security
A10.1.2	Structured systems	Cryptography security
A11.1.1	Structured work	Physical security
A11.1.1	Structured systems	Physical security
A11.1.2	Structured systems	Physical security
A11.1.3	Structured work	Physical security
A11.1.3	Structured systems	Physical security
A11.1.4	Structured systems	Physical security
A11.1.4	Structured work	Physical security
A11.1.5	Structured work	Physical security
A11.1.5	Structured systems	Physical security
A11.1.6	Structured systems	Physical security
A11.2.1	Structured systems	Physical security
A11.2.2	Structured systems	Physical security
A11.2.3	Structured systems	Physical security
A11.2.4	Structured systems	Physical security
A11.2.5	Structured work	Physical security
A11.2.6	Structured systems	Physical security
A11.2.7	Structured work	Physical security
A11.2.8	Structured work	Physical security
A11.2.9	Structured work	Physical security
A12.1.1	Structured work	Security awareness
A12.1.2	Structured work	Security awareness
A12.1.3	Structured work	Security awareness
A12.1.4	Structured systems	Configuration security
A12.2.1	Structured work	Security awareness
A12.2.1	Structured systems	Configuration security
A12.2.1	Structured work	Human error
A12.3.1	Structured work	Configuration security
A12.4.1	Structured work	Security awareness

A12.4.1	Structured systems	Configuration security
A12.4.2	Structured systems	Physical security
A12.4.3	Structured work	Security awareness
A12.4.3	Structured work	Physical security
A12.4.4	Structured systems	Configuration security
A12.5.1	Structured environment	Configuration security
A12.6.1	Structured environment	Configuration security
A12.6.1	Structured work	Security awareness
A12.6.1	Structured systems	Configuration security
A12.6.1	Structured systems	Software security
A12.6.2	Structured work	Configuration security
A12.7.1	Structured work	Security awareness
A13.1.1	Structured work	Configuration security
A13.1.1	Structured work	Security awareness
A13.1.2	Structured work	Configuration security
A13.1.3	Structured systems	Configuration security
A13.2.1	Structured systems	Configuration security
A13.2.1	Structured work	Security awareness
A13.2.2	Structured work	Supply chain security
A13.2.3	Structured systems	Configuration security
A13.2.4	Structured work	Security awareness
A14.1.1	Structured work	Security awareness
A14.1.2	Structured systems	Cryptography security
A14.1.3	Structured systems	Cryptography security
A14.2.1	Structured work	Software security
A14.2.2	Structured work	Configuration awareness
A14.2.3	Structured work	Configuration awareness
A14.2.4	Structured work	Software security
A14.2.5	Structured work	Security awareness
A14.2.6	Structured work	Security awareness

A14.2.6	Structured systems	Security awareness
A14.2.7	Structured work	Software security
A14.2.8	Structured work	Configuration security
A14.2.9	Structured work	Configuration security
A14.3.1	Structured systems	Software security
A15.1.1	Structured work	Supply chain security
A15.1.2	Structured work	Supply chain security
A15.1.3	Structured work	Supply chain security
A15.2.1	Structured work	Supply chain security
A15.2.2	Structured work	Supply chain security
A16.1.1	Structured work	Security awareness
A16.1.2	Structured work	Security awareness
A16.1.3	Structured work	Security awareness
A16.1.4	Structured work	Security awareness
A16.1.5	Structured work	Security awareness
A16.1.6	Structured work	Security awareness
A16.1.7	Structured work	Security awareness
A17.1.1	Structured work	Security awareness
A17.1.2	Structured work	Security awareness
A17.1.2	Structured systems	Security awareness
A17.1.3	Structured work	Security awareness
A17.1.3	Structured work	Configuration security
A17.2.1	Structured systems	Configuration security
A17.2.1	Structured work	Security awareness
A18.1.1	Structured systems	Configuration security
A18.1.1	Structured work	Security awareness
A18.1.2	Structured work	Security awareness
A18.1.3	Structured work	Physical security

A18.1.3	Structured work	Configuration security
A18.1.3	Structured work	Physical security
A18.1.4	Structured work	Configuration security
A18.1.5	Structured work	Cryptography security
A18.2.1	Structured work	Security awareness
A18.2.2	Structured work	Security awareness
A18.2.3	Structured systems	Configuration security

Appendix 5. Benchmarking of controls in CIS controls

This table contains mapping of each control defined in CIS controls against our framework, 196 mappings in total. Class names have been truncated.

Table 2. Mappings from CIS controls to the framework presented in this thesis.

Control ID	**Target**	**Class**
1.1	Structured work	Security awareness
1.2	Structured work	Security awareness
1.3	Structured systems	Security awareness
1.4	Structured work	Security awareness
1.5	Structured work	Security awareness
1.6	Structured systems	Security awareness
1.6	Structured work	Security awareness
1.7	Structured systems	Security awareness
1.8	Structured systems	Security awareness
2.1	Structured work	Security awareness
2.2	Structured work	Security awareness
2.2	Structured work	Software security
2.3	Structured work	Security awareness
2.4	Structured systems	Security awareness
2.5	Structured work	Security awareness
2.6	Structured work	Security awareness
2.7	Structured work	Security awareness
2.8	Structured systems	Security awareness
2.9	Structured systems	Security awareness
2.10	Structured systems	Security awareness
3.1	Structured work	Configuration security
3.2	Structured work	Configuration security
3.3	Structured systems	Configuration security

3.4	Structured systems	Configuration security
3.4	Structured systems	Software security
3.5	Structured systems	Configuration security
3.5	Structured systems	Software security
3.6	Structured work	Software security
3.7	Structured work	Software security
4.1	Structured work	Configuration security
4.2	Structured systems	Configuration security
4.3	Structured systems	Configuration security
4.4	Structured systems	Configuration security
4.5	Structured systems	Configuration security
4.5	Structured systems	Cryptography security
4.6	Structured systems	Configuration security
4.7	Structured systems	Configuration security
4.8	Structured systems	Configuration security
4.9	Structured systems	Configuration security
5.1	Structured work	Software security
5.1	Structured work	Configuration security
5.2	Structured work	Configuration security
5.3	Structured work	Configuration security
5.3	Structured systems	Configuration security
5.4	Structured systems	Configuration security
5.5	Structured systems	Configuration security
6.1	Structured systems	Configuration security
6.2	Structured systems	Configuration security
6.3	Structured systems	Configuration security
6.4	Structured systems	Configuration security
6.5	Structured systems	Configuration security
6.5	Structured work	Security awareness

6.6	Structured systems	Configuration security
6.7	Structured work	Security awareness
6.8	Structured systems	Configuration security
6.8	Structured work	Security awareness
7.1	Structured work	Software security
7.2	Structured systems	Software security
7.3	Structured systems	Software security
7.4	Structured systems	Configuration security
7.5	Structured work	Security awareness
7.5	Structured systems	Configuration security
7.6	Structured work	Security awareness
7.6	Structured systems	Configuration security
7.7	Structured systems	Configuration security
7.8	Structured systems	Configuration security
7.9	Structured systems	Configuration security
7.10	Structured systems	Configuration security
8.1	Structured systems	Security awareness
8.2	Structured work	Configuration security
8.3	Structured systems	Configuration security
8.4	Structured systems	Configuration security
8.5	Structured systems	Configuration security
8.6	Structured systems	Security awareness
8.7	Structured systems	Security awareness
8.8	Structured systems	Security awareness
9.1	Structured work	Configuration security
9.2	Structured work	Configuration security
9.3	Structured work	Configuration security
9.4	Structured systems	Configuration security
9.5	Structured systems	Configuration security

10.1	Structured systems	Security awareness
10.2	Structured systems	Security awareness
10.3	Structured work	Security awareness
10.4	Structured systems	Physical security
10.4	Structured systems	Cryptography security
10.4	Structured systems	Configuration security
10.5	Structured systems	Configuration security
11.1	Structured work	Security awareness
11.2	Structured work	Security awareness
11.3	Structured work	Configuration security
11.4	Structured systems	Configuration security
11.4	Structured systems	Software security
11.5	Structured systems	Configuration security
11.6	Structured systems	Configuration security
11.6	Structured work	Security awareness
11.7	Structured systems	Configuration security
12.1	Structured work	Security awareness
12.2	Structured work	Configuration security
12.3	Structured systems	Configuration security
12.4	Structured systems	Configuration security
12.5	Structured systems	Security awareness
12.6	Structured systems	Security awareness
12.6	Structured systems	Configuration security
12.7	Structured systems	Configuration security
12.8	Structured systems	Security awareness
12.9	Structured systems	Configuration security
12.10	Structured systems	Cryptography security
12.11	Structured systems	Configuration security
12.11	Structured systems	Cryptography security

12.12	Structured systems	Security awareness
13.1	Structured work	Security awareness
13.2	Structured work	Configuration security
13.3	Structured systems	Security awareness
13.4	Structured work	Configuration security
13.5	Structured systems	Security awareness
13.6	Structured systems	Configuration security
13.7	Structured systems	Configuration security
13.8	Structured systems	Configuration security
13.9	Structured systems	Cryptography security
14.1	Structured systems	Configuration security
14.2	Structured systems	Configuration security
14.3	Structured systems	Configuration security
14.4	Structured systems	Cryptography security
14.5	Structured work	Security awareness
14.6	Structured work	Configuration security
14.6	Structured work	Configuration security
14.7	Structured systems	Configuration security
14.8	Structured systems	Cryptography security
14.9	Structured systems	Security awareness
15.1	Structured work	Security awareness
15.2	Structured systems	Configuration security
15.3	Structured systems	Configuration security
15.4	Structured systems	Configuration security
15.5	Structured systems	Configuration security
15.6	Structured systems	Configuration security
15.7	Structured systems	Cryptography security
15.8	Structured systems	Configuration security
15.9	Structured systems	Configuration security

15.10	Structured systems	Configuration security
16.1	Structured work	Security awareness
16.2	Structured systems	Configuration security
16.3	Structured systems	Configuration security
16.4	Structured systems	Cryptography security
16.5	Structured work	Cryptography security
16.6	Structured work	Configuration security
16.7	Structured work	Configuration security
16.8	Structured systems	Configuration security
16.9	Structured systems	Configuration security
16.10	Structured work	Configuration security
16.11	Structured systems	Configuration security
16.12	Structured work	Security awareness
16.13	Structured systems	Security awareness
17.1	Structured work	Security awareness
17.2	Structured work	Security awareness
17.3	Structured work	Security awareness
17.4	Structured work	Security awareness
17.5	Structured work	Security awareness
17.6	Structured work	Security awareness
17.7	Structured work	Security awareness
17.7	Structured work	Human error
17.8	Structured work	Security awareness
17.8	Structured work	Human error
17.9	Structured work	Security awareness
18.1	Structured work	Software security
18.2	Structured work	Software security
18.3	Structured work	Software security
18.3	Structured environment	Software security

18.4	Structured work	Software security
18.5	Structured work	Cryptography security
18.6	Structured work	Software security
18.7	Structured systems	Software security
18.8	Structured work	Software security
18.9	Structured systems	Configuration security
18.10	Structured systems	Configuration security
18.11	Structured work	Configuration security
19.1	Structured work	Security awareness
19.2	Structured work	Security awareness
19.3	Structured work	Security awareness
19.4	Structured work	Security awareness
19.5	Structured environment	Security awareness
19.6	Structured work	Security awareness
19.7	Structured work	Security awareness
19.8	Structured work	Security awareness
20.1	Structured work	Security awareness
20.2	Structured work	Security awareness
20.3	Structured work	Security awareness
20.3	Structured work	Software security
20.3	Structured work	Configuration security
20.4	Structured work	Software security
20.4	Structured work	Security awareness
20.4	Structured work	Configuration security
20.5	Structured systems	Configuration security
20.6	Structured work	Software security
20.6	Structured work	Configuration security
20.6	Structured work	Security awareness
20.7	Structured work	Security awareness
20.8	Structured work	Configuration security