Freie Universität Berlin

Development, Implementation and Evaluation of a Security Concept for the Secure Transfer of Sensitive Personal Data Between Different Devices

*A **thesis** presented for the degree of **Master of Science**.*

Oliver Junk, Freie Universität Berlin, Germany

Matriculation number: 4568642

oliver132@zedat.fu-berlin.de

November 18, 2019

Supervisor:

Nicolas Lehmann[1], Freie Universität Berlin, Germany

Reviewers:

Prof. Dr. Agnès Voisard[2], Freie Universität Berlin, Germany

Prof. Dr. Marian Margraf[3], Freie Universität Berlin, Germany

Citation:

Oliver Junk, *Development, Implementation and Evaluation of a Security Concept for the Secure Transfer of Sensitive Personal Data Between Different Devices*, Freie Universität Berlin, Master Thesis, 2019

Version: Final Version

[1]Dept. of Computer Science and Mathematics, Databases and Information Systems Group
[2]Dept. of Computer Science and Mathematics, Databases and Information Systems Group
[3]Dept. of Computer Science and Mathematics, ID-Management Group

Statutory Declaration

I declare that I have developed and written the enclosed Master thesis completely by myself, and have not used sources or means without declaration in the text. Any thoughts from others or literal quotations are clearly marked.

The Master thesis was not used in the same or in a similar version to achieve an academic grading or is being published elsewhere.

Berlin (Germany), November 18, 2019

_____

Oliver Junk

Zusammenfassung

In den letzten Jahren hat der Gesundheitssektor Fortschritte bei der Digitalisierung seiner Prozesse und Ressourcen erzielt. Mit den Fortschritten in der Kommunikationstechnologie kamen allerdings auch ihre Nachteile: Elektronische Informationssysteme bieten Angriffsfläche für bösartige Akteure. Die Europäische Union hat die Datenschutz-Grundverordnung (DSGVO) in Kraft gesetzt, um die Privatsphäre ihrer Bürger zu schützen. In dieser Masterarbeit wird ein Sicherheitskonzept zur Sicherung der Weitergabe personenbezogener Daten und zur Einhaltung der Datenschutz-Grundverordnung für den Einsatz im CliniScale Projekt entworfen. Die Forschung beginnt mit einer systematischen Literaturrecherche und einer Analyse der DSGVO, um gesetzliche Anforderungen zu identifizieren, denen das in dieser Masterarbeit entwickelte Konzept entsprechen muss. Nach der Modellierung des CliniScale Systems wird eine Risikoanalyse zur Bewertung des Sicherheitsrisikos mithilfe der Modular Risk Assessment (MoRA) Methodik durchgeführt, um kritische Infrastrukturen zu identifizieren und Sicherheitsmaßnahmen zu implementieren, die in Betracht der aufgetretenen Risiken angemessen sind. Dabei wird die Sicherheit des erstellten Konzepts validiert. Für die identifizierten Sicherheitsmaßnahmen wird eine Implementierungsrichtlinie erstellt, die es ermöglicht, das Sicherheitskonzept nicht nur in der Umgebung des CliniScale Projekts, sondern für jede ähnliche Infrastruktur im Gesundheitswesen einzusetzen.

Abstract

In recent years, the healthcare sector made advances in digitalizing their processes and resources. With advances in communication technology, also came their drawbacks: Electronic information is pruned to being exploited by adversaries. The European Union implemented the General Data Protection Regulation (GDPR) in order to protect the privacy of their citizen. In this thesis, a security concept securing the transfer of personal data and complying with the GDPR is designed for the use in the CliniScale project. The research starts with a systematic literature research and an analysis of the GDPR in order to identify regulations this thesis has to comply with. After modeling the CliniScale system environment, a security risk assessment is performed using the Modular Risk Assessment (MoRA) methodology to identify critical infrastructure and implement security measurements appropriate to the encountered risks. In the process, the security of the created concept is validated. An implementation guideline is created for the identified security measurements, enabling the implementation of the security concept not only in the environment of the CliniScale project, but for every similar infrastructure in the healthcare domain.

Table of Contents

List of Tables

List of Figures

List of Source Codes

1. Introduction

This first chapter, *Introduction*, consists of three sections. The first section, *Motivation*, presents issues regarding the communication and transportation of sensitive personal data. The second section, *Outline of Contribution*, defines how this master thesis contributes to solve these issues by designing a security concept for the secure transfer of sensitive data. The third section, *Structure of the Thesis*, gives an overview on how this master thesis is organized.

1.1. Motivation

Digital communication and information resources affect almost every aspect of our lives. Personal information is collected everywhere as persons give it up willingly by using platforms such as social media platforms or bonus programs while shopping. With the increased availability of information in electronic form, issues concerning data privacy surface, such as the Facebook Cambridge Analytica[1] affair or large-scale data breaches in different companies show. Big corporations fail or intentionally neglect to protect their customers privacy to pursue profit.

Personal data is any data of an identifiable individual, such as name, date of birth and gender, but also sensitive data such as ethnicity, sexual orientation and religious association. Disclosure of this data can lead to great harm to an individual. From a declined job interview because of different political beliefs, increased health insurance coverage to threats such as blackmailing and racial persecution.

To protect the customers right to privacy, the European Union established the *General Data Protection Regulation*[33], a law on data protection and privacy for all individuals living in the European Union. Its goal is to provide the customers with more control over their data by unifying the regulations in the European Union. This regulation demands of controllers of personal data to install and maintain technical and organizational measures to comply the data protection principles, such as using pseudonymization and to use the highest possible privacy settings by default.

The GDPR defines a special category of personal data: sensitive personal data[18]. It consists of data revealing political opinions, racial or ethnic origins, religious or philosophical beliefs, medical information and more. Processing of sensitive personal data demands compliance with stricter regulations to further protect the privacy of an individual.

Example 1 *Mr. Ronnie Coleman is taking part in a clinical trial. For this he regularly shoots photos of his skin with a special camera after applying the skin cream to be tested. In these photos a skin disease of Mr. Coleman is visible, which he*

[1] The Guardian; The Cambridge Analytica Files: https://www.theguardian.com/news/series/cambridge-analytica-files. (Online; last accessed: November 18, 2019)

wants to keep private. One day, Mr. Coleman sends the photos from an unsecured network. The data is intercepted by a malicious hacker. The hacker blackmails Mr. Coleman with the publication of the photos. Mr. Coleman agrees to the demands of the blackmailer for fear of losing his job or being socially marginalized.

The security and privacy of personal data, especially those that fall under the category of sensitive personal data, is essential to guarantee an individual's right of self-determination. The lack or bad implementation of a security concept to protect sensitive data often leads to a breach of regulations. A well thought security concept using up-to-date cryptographic protocols ensuring critical properties such as confidentiality and integrity is needed to protect the security and privacy of individuals.

1.2. Outline of Contribution

The goal of this Master thesis is to design a security concept for the transfer of sensitive personal data complying with the GDPR, evaluate its security level and give an implementation guideline to identified security measurements.

As part of the CliniScale project[2], the concept is primarily designed for the secure communication of medical data. Since there is not made any distinction between different kind of data under the category of sensitive personal data, the concept designed is universally applicable to any transfer of sensitive personal data.

The GDPR is analyzed in order to identify requirements to the security concept designed in this thesis to ensure the concepts compliance with the regulation. The process of the CliniScale system is analyzed in order to model it as a preliminary task for the security risk assessment. A technical guideline to counsel for the choice of state-of-the-art cryptographic protocols and their configuration is identified in order to meet the requirements of the GDPR.

The security concept is designed performing a security risk assessment by applying the MoRA methodology. The methodology is altered in order to support the Microsoft Threat Modeling Tool as a threat and control catalogue.

Further, a guideline to implement security measurements identified in the process of the security risk assessment is created.

[2] CliniScale Project: http://www.mi.fu-berlin.de/en/inf/groups/ag-db/projects/CliniScale/index.html. (Online; last accessed: November 18, 2019)

2 **Oliver Junk**, *Development, Implementation and Evaluation of a Security Concept for the Secure Transfer of Sensitive Personal Data Between Different Devices*, 2019

1.3. Structure of the Thesis

Introduction The chapter *Introduction* covers the three sections *Motivation*, *Outline of Contribution* and *Structure of the Thesis*. The section *Motivation* describes the current situation and issues regarding privacy and security in a data driven world. Afterwards, the section *Outline of Contribution* presents the work done as part of this thesis, the design and validation of a security concept to secure the communication in a domain were data privacy is of critical importance. At last, the section *Structure of the Thesis* gives an overview and description of every chapter and their sections this thesis contains.

Background The chapter *Background* consists of the four sections *Definitions*, *Related System*, *CliniScale Project* and *Related Works*. The first section *Background* gives an explanation and definition of terms and concepts which are used in this thesis. The second section, *Related System*, presents a project with similarities to the CliniScale system and identifies security measurements implemented. The third section, *CliniScale Project*, presents the project this thesis is part of. The last section, *Related Works*, presents academic work in the domain of this thesis.

Methodology: Modular Risk Assessment This chapter presents the methodology used in this thesis to perform a security risk assessment and consists of the four sections *Work Products*, *Activities*, *Artifacts* and *Modifications*. The first section, *Work Products*, defines the artifacts created by the activities. Section two, *Activities*, gives detail about the workflows to create work products. The third section, *Artifacts*, describes external tools that help in performing a risk assessment. The fourth section, *Modifications*, explains the alterations made to the MoRA methodology in order to integrate the Microsoft STRIDE threat model and the Microsoft Threat Modeling Tool.

Analysis The chapter *Analysis* provides examination in different domains regarding this thesis. It consists of the three sections *General Data Protection Regulation*, *CliniScale System Process* and *Technical Guidelines for Cryptographic Protocols*. The first section, *General Data Protection Regulation*, analyzes the GDPR and identifies legal restraints this thesis has to comply with. The section *CliniScale System Process* analyzes and models the process of the CliniScale system. The third section, *Technical Guidelines for Cryptographic Protocols*, identifies technical guidelines to consult when implementing cryptographic protocols.

Architecture Design and Security Risk Assessment In the chapter *Architecture Design and Security Risk Assessment* the security risk assessment is performed. It consists of the sections *Assessment Model*, *Document System Under Development* and *Security Risk Assessment*. The section *Assessment Model* defines the Assessment Model as a preliminary task for the risk assessment. The second section, *Document System Under Development*, describes how the CliniScale system is modeled as System Under Development and imported in the Microsoft Threat Modeling Tool.

The third section, *Security Risk Assessment*, describes the performance of the security risk assessment.

Implementation The chapter *Implementation* creates a guideline to implement identified controls into the existing CliniScale implementation. The chapter consists of the three sections *CliniScale System Architecture*, *Technologies* and *Consequences for the Implementation*. The first section, *CliniScale System Architecture*, analyzes the current implementation of the CliniScale project. The second section, *Technologies*, presents the frameworks and libraries used in the implementation. The third section, *Consequences for the Implementation*, presents a guideline on how to implement the controls identified in the risk assessment into the CliniScale implementation.

Results This chapter consists of the two sections *Summary* and *Limitations* and presents the result of this thesis. The section *Summary* lists the findings of this thesis. The second section, *Limitations*, details aspects of this thesis that have not been examined in this work.

Conclusion The last chapter *Conclusion* consists of two sections: *Discussion of Results* and *Future Work*. The section *Discussion of Results* opens a debate about the results found in this thesis. The section *Future Work* presents possible suggestions for future research following the work made in this thesis.

2. Background

The chapter *Background* consists of the four sections *Definitions*, *Related System*, *CliniScale Project* and *Related Works*. The first section, *Definitions*, explains basic terms to understand the topic of this thesis. The second section, *Related System*, presents an application similar to the CliniScale project. The third section, *CliniScale Project*, presents the project this thesis is part of. The fourth section, *Related Works*, analyzes the current scientific landscape related to this thesis.

2.1. Definitions

This section defines technical terms needed to understand this thesis.

CIA-Triad A security model for the development of security policies within an organization, originally consisting of confidentiality, integrity and availability. The CIA-Triad is extended by three additional security properties: accountability, authorization and authenticity. **Confidentiality** is part of privacy. It describes the property that information is not made available or disclosed to unauthorized individuals, entities or processes. **Integrity** describes the property of maintaining consistency, accuracy and trustworthiness of data. **Availability** describes the property that authorized parties are able to access resources when needed. **Accountability** describes the security property of non-repudiation of every party to their actions. It prevents a party of denying performing an action, such as receiving or sending a transaction. **Authorization** defines the property defining if a party is allowed to access a resource after authentication. **Authenticity** describes the property of proper verification of a party's identity.

General Data Protection Regulation (GDPR) The General Data Protection Regulation is a regulation in European Union (EU) law on data protection and privacy for all individuals within the EU and the European Economic Area implemented May 25, 2018[33]. The aim of the GDPR is to protect all EU citizens from privacy and data breaches.

Sensitive Personal Data Personal data which are by their nature particularly sensitive in relation to fundamental rights and freedoms are defined as sensitive personal data[18]. They merit specific protection as the context of their processing could create significant risks to fundamental rights.

Modular Risk Assessment (MoRA) The MoRA methodology is a security risk assessment methodology[26]. MoRA offers a risk-driven, iterative and modular approach and offers guidance and catalogs for practical application. The method is defined in chapter 3.

Microsoft Threat Modeling Tool [35] The Microsoft Threat Modeling Tool is a core element of the Microsoft Security Development Lifecycle. The tool allows software architects to identify and mitigate security issues early in the process of designing a software architecture. It is based on data flow diagrams to model the architecture. Threats get identified using the STRIDE threat classification scheme. The tool also consists of a security control catalogue, offering information on how to mitigate found threats.

Microsoft STRIDE STRIDE is a threat model developed by Praerit Garg and Loren Kohnfelder at Microsoft[34]. The model helps identifying possible threats by categorizing them in six categories: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privileges.

Yakindu Security Analyst [1] The Yakindu Security analyst is a tool created by Itemis AG[2] to perform risk assessments. It offers a highly customizable environment making it usable for different risk assessment methodologies.

2.2. Related System

This section presents a comparable application to the CliniScale system that exchanges sensitive personal data in a client server infrastructure using a mobile application.

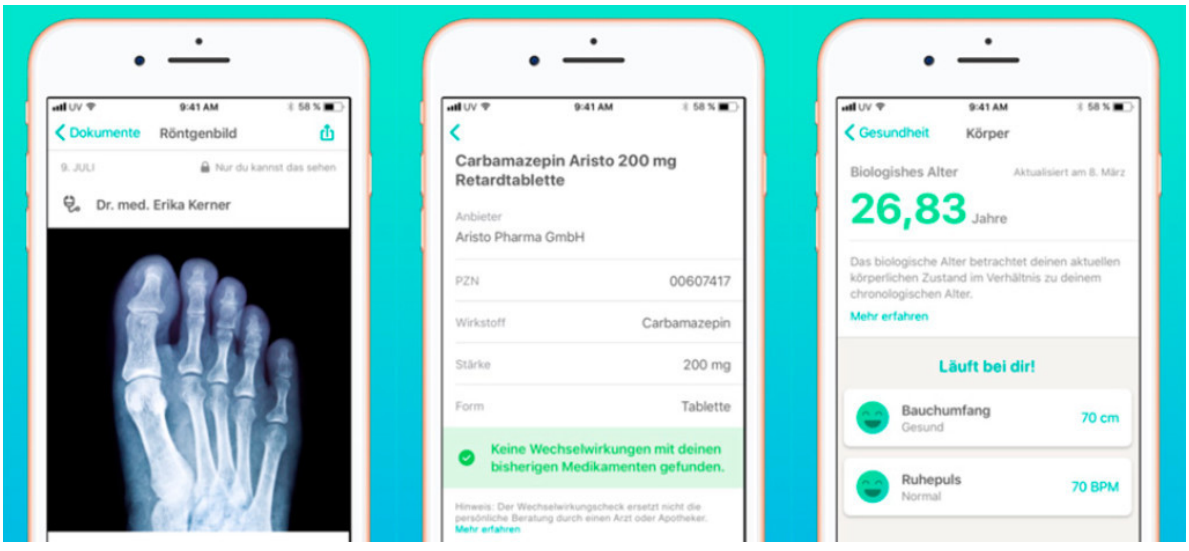

Figure 2.1.: Vivy App

[1] Yakindu Security Analyst: https://www.itemis.com/en/yakindu/security-analyst/. (Online; last accessed: November 18, 2019)
[2] Itemis AG: https://www.itemis.com/en/. (Online; last accessed: November 18, 2019)

Vivy[3] is an application made by the German Vivy GmbH allowing users to manage their electronic health records. The security aspects of the app are reviewed by the Secure Systems Engineering department of the Fraunhofer AISEC in a whitepaper[36].
Vivy follows the recommendations of the German Federal Office for Information Security (BSI) by complying to standards such as the BSI 200-1 and BSI 200-2. Vivy uses cryptographic protocols and cipher suites recommended by the BSI in their technical guidelines BSI TR-02102-1 and BSI TR-02102-2.
Below are some of their implemented security features that are of interest regarding this thesis:

Two-factor Authentication Vivy uses two-factor authentication for all its users. Beside the user made password the user's smart phone is linked to his account.

Data Encryption All data saved on Vivy servers are encrypted with a user specific key, meaning that only the user can access the data using their private key. Vivy uses a hybrid encryption scheme, encrypting the data using a symmetric cryptographic technique, AES in GCM mode using a 256-bit key. RSA using OAEP padding and 4096-bit keys is used to exchange the symmetric keys.

Network Security Vivy secures the communication and data exchange between its users by encrypting the data transferred and authenticating the communication partner. TLS 1.2 is used in the form of ELBSecurityPolicy-TLS-1-2-2017-01.

2.3. CliniScale Project

CliniScale is a project by the Databases and Information Systems Group[4] at Freie Universität Berlin with the goal to run clinical trials in the population. Combining web services with mobile applications enables clinical test centers to run scalable, user-friendly clinical trials in the population.
The main concept is to make clinical trials feasible in the population using mobile devices. The project provides solutions for the creation of clinical trials, proband selection procedures, executing the trial on a mobile device by selected probands and evaluation and presentation of the trial results.
Of interest for this thesis are the process, defined in section 4.2, and the current implementation of the CliniScale back end and the Android mobile application, defined in section 6.1.

[3] Vivy: https://www.vivy.com/. (Online; Last accesses November 18, 2019
[4] Databases and Information Systems: https://www.mi.fu-berlin.de/en/inf/groups/ag-db/index.html. (Online; last accessed: November 18, 2019)

2.4. Related Works

Related Work provides an overview on related scientific work, academic research and books covering the topics of *Security Risk Assessment*, *Secure Communication* and *Privacy in Healthcare*.

2.4.1. Security Risk Assessment

[26] by Jörn Eichler and Daniel Angermeier of the Fraunhofer AISEC presents the MoRA method to perform a security risk assessment. An in-depth explanation of this method can be found in the chapter Methodology: Modular Risk Assessment. [20] extends the original MoRA method on how to systematically identify security goals and threats. [19] further builds on the methodology by improving the identification, merging and validation of security goals.

[28] describes the role of threat and control catalogues in a security risk assessment. The actual and perceived qualitative and quantitative effect of using catalogues as part of a security risk assessment is researched. The quantitative analysis shows that non-security experts using catalogues reach results comparable to experts that are not using catalogues, while the perceived ease of use was slightly lower. It is also found out that security experts and non-experts have different expectations towards the catalogues. While experts are more interested in a common terminology and a checklist to validate that nothing was left out, non-experts are more interested in the ease of navigating through the catalogue, refusing larger and less specific catalogues.

In [38] the effectiveness of the Microsoft Threat Modeling Tool is evaluated by performing a course assignment on computer science graduates. The graduates performed two thread modeling tasks, first using a manual process and then using the threat modeling tool. Results show that the usage of the Microsoft Threat Modeling Tool improved the quality of the graduates work. The abstraction level of the defined model affects the amount of threats identified up to a point, were the number of generated threats is being limited.

2.4.2. Secure Communication

[29] presents an analysis of the TLS protocol and its application to data encryption. It shows how key encapsulation mechanisms can be extracted from the protocol and how the security of the entire protocol depends on the key encapsulation. This work suggests using CCA-secure implementations of the TLS protocol, as side channel attacks are possible on other variants.

In [21] the authors present a formalism for the analysis of key exchange protocols. Their findings are that any key exchange protocol can be paired with symmetric encryption and authentication functions in order to provide secure communication channels. Secure communication channels are a fundamental aspect of this thesis and Canetti and Krawczyk made fundamental groundwork in defining these and proving their security.

In [27] the authors inspect the implementation of SSL/TLS protocols in android mobile applications by introducing MalloDroid, a tool to detect potential vulnerabilities against Man-In-The-Middle (MITM) attacks. 13500 popular applications have been investigated and 1074 revealed to be potentially vulnerable to MITM attacks. As a consequence, measurements categorized in three categories, OS solutions, App market solutions and Standalone solutions, are presented. The OS solutions are of interest. They recommend certificate pinning, usage of HTTPS in every connection, improved permission policies and visual security feedback.

[25] describes various ways of storing secret keys in a secure way on Android, analyzing both software and hardware solutions. A critical part of successfully implementing cryptography is to ensure a secure way to store the needed secret keys. The authors work describes different solutions and discusses their advantages, making it a good source for this thesis to base its design decisions on.

2.4.3. Privacy in Healthcare

In [37] the authors perform a study researching two aspects of using medical assistive technologies, security and privacy. There is a focus laid on the perceived importance of these aspects by users. Results show that, while healthy adults insist on the highest security and privacy standards, elderly and adults with ailing health attributes have a perceived lower importance towards security and privacy since they value fast and continuous help and monitoring higher. This shows that people, especially those in greater need, are ready to accept cuts into their security and privacy for medical assistance, something that the pharmacy industry or governments may want to exploit. From a security standpoint of view, it shows that it is of utmost importance to guarantee high level of security and privacy standards to everybody, especially those who are willing to give them up for medical assistance and risk being exploited.

In [31] the authors study the current technological advances in the health care sector and its impact on security and privacy. Additionally, existing methods to handle these issues are presented. Many issues regarding health data are discussed, such as data access and storage, and solutions to these problems are presented and are of interest for this thesis. Authentication and encryption are required to comply with the GDPR.

[24] presents the authors research on medical data mining and possible security and privacy issues. Topics discussed involve privacy-conserving medical data mining and ownership of medical data, both interesting for this thesis. Data anonymization is

found to be a key element in preserving privacy while still allowing researchers access to the medical data. Additionally, the ”right to explanation” of the GDPR is found to be in conflict with deep learning algorithms and neural networks, as often the logic found by them are black boxes that lack transparency when it comes to understanding how they arrived at their decisions.

[23] perform a study examining users attitude towards security and privacy for mobile devices and how it differs from the user’s usage of traditional devices such as laptops and PCs. The survey shows, that generally, users are more concerned about security and privacy when using their phones. A large number of the participants do not perform privacy critical tasks on their phones. When the participants were asked to rationally explain their decisions, the main problems regarding security on a mobile devices can be traced back to lack of trust into communication technologies, such as Wi-Fi or 3G, mobile devices lack of anti-virus protection, lack of knowledge about mobile devices and the easiness to lose a mobile phone. Participants that were more concerned about security and privacy on laptops and PCs noted that they do not perform privacy critical tasks, such as banking or entering their social security number, on their mobile phone at all. Understanding the user’s motives and needs is an important factor in designing a privacy critical application. Complying and using international accepted standards may help improve the users trust into an application, making it viable to a larger audience. The authors present recommendations on how to improve user’s confidence towards security and privacy of a mobile application.

3. Methodology: Modular Risk Assessment

The chapter *Methodology: Modular Risk Assessment* presents the methodology used to perform the core part of this thesis: the security risk assessment.

Modular Risk Assessment (MoRA) is a method for security risk assessment developed by Prof. Dr. Eichler and Dr. Angermeier from the Fraunhofer AISEC in 2015. Originally designed to be used in the automotive domain, its core objective flexibility of application in different environments enables the application of MoRA in other domains as well. MoRA further features a modular structure, a unified method framework, well-defined artifacts as interfaces between activities and various guidelines.

The MoRA method consists of various activities and work products, supported by a set of guidelines and artifacts. The following sections *Work Products*, *Activities* and *Artifacts* present each of these core elements.

3.1. Work Products

MoRA consists of two work products: *System Under Development* and *Risk Assessment*. They describe the artifacts created when performing the processes defined under *Activities*.

3.1.1. System Under Development

The *System Under Development* (*SUD*) represents the basic architecture of the underlying system. The *SUD* consists of various elements which can be either functions, components, data or data channels. Functions characterize the *SUDs* functionality and describe the main tasks the system must be able to perform. Components represent the basic physical elements of every system, such as servers, databases, mobile phones. Data describes the information that may be produced or processed in a component and is transferred over data channels. Data channels are the means of transportation of data between different components. A data channel consists data flows, each having a source and a destination component and transferred data elements. Functions, components and data elements may be decomposed hierarchically in order to address the required level of detail.

3.1.2. Risk Assessment

The *Risk Assessment* documents the results of the activities *Determine Protection Needs*, *Analyze Threats*, *Analyze Risks* and *Establish Controls*. It consists of security goals, threats, risks and controls.

Security Goals describe the main protection needs of a system and consist of a referenced function, component or data channel as well as a security goal class. Asserted damage criteria define the damage potential of a potential breach of the security goal. Security goals can be linked hierarchically, indicating their dependencies and allowing for propagation of their damage potentials.
Threats describe major endangerments of security goals. Every threat instantiates a threat class of the threat catalogue. They reference components or data channels and security goals they have an impact on. Threats are also rated by a required attack potential, describing the skill level of an adversary needed to perform the threat.
Risks combine security goals and matching threats and controls, forming a risk potential, which is calculated by aggregating the damage potential of the security goals and the attack potential of threats reduced by given controls. Risk elements describe the overall risk level of the risk assessment.
Controls describe security measurements to protect security goals by mitigating respective threats. Every control instantiates a control class, contains references to mitigated threats and an defines a required attack potential. The required attack potential of threats is overwritten by these, usually, lower values, creating a new required attack potential. Controls can introduce new components and respective security goals.

3.2. Activities

The MoRA method defines various activities, entailing a hierarchy of tasks that define purpose, input and output in terms of work products as well as a set of steps to execute. The activities are *Document System Under Development*, *Determine Protection Needs*, *Analyze Threats*, *Analyze Risks* and *Establish Controls*.

3.2.1. Document System Under Development

This activity is responsible for creating the *System Under Development*. First, the functions of the system are identified and decomposed into a hierarchic structure. The main functions of the system may be decomposed into various sub-functions, such as a function that retrieves the information and another function that processes it. Second, the investigated systems architecture is modeled out of components, data and data channels.
Experience of the security expert helps to determine the needed granularity of the created model. Going into detail without gaining any improvement regarding security can impede the whole process.

3.2.2. Determine Protection Needs

Goal of this activity is to identify and determine security goals and estimate the damage potential resulting from a violation of these. The decomposition of the *SUD* helps identifying security goals, by matching its elements with security goal classes. For each of these potential security goals, a list of damage criteria from the *Assessment Model* is checked. If any damage criteria are applicable, a security goal is found. The damage potential of the security goal is determined by the aggregated damage potentials of the assigned damage criteria.

3.2.3. Analyze Threats

This activity identifies potential threats to the security goals determined in the previous activity. To identify applicable threats, all previously determined security goals and the class of their referenced element in the *SUD* are paired with the threat catalogue. If the security goal class and the specific element class match, a potential threat is found. The threats required attack potential is estimated by defining a combination of risk factors.

3.2.4. Analyze Risks

The activity *Analyze Risks* identifies risks by evaluating threats declared in the beforehand activity. Threats pose a required attack potential, rating the difficulty to perform the given threat. This rating combined with the aggregated damage potentials of affected security goals, form a risk level. Risks can contain various security goals and threats and allow for a fast overview of the current security level of the investigated system.

3.2.5. Establish Controls

The previous activity, *Analyze Risks*, results in a list of risks with a risk level rating. If the computed rating does not match the targeted lowest acceptable risk level, measurements to mitigate the threats of the risk must be established. To identify the right control, for every threat associated with the risk the control catalogue is searched for a control class which matches the *SUD* element type and the threat class. Out of the found control classes, one or more can be implemented in the system to mitigate the threat. The risk element should now compute a new, usually lower, risk level. Beware that some controls introduce new elements to the *SUD*, such as security keys when implementing cryptographic protocols. These elements also introduce new security goals, such as these security keys must be kept confidential. This means that, if new elements and security goals are introduced, all activities may have to be iterated over again until no new elements are introduced.

3.3. Artifacts

Artifacts help in performing the MoRA method by complementing activities and work products. These consist of an *Assessment Model*, the *Threat Catalogue* and the *Control Catalogue*.

3.3.1. Assessment Model

The *Assessment Model* defines various variables needed to perform the MoRA methodology. Security Goal Classes define different type of security properties. For further information, see *Definitions*. Damage Potentials estimate the expected damage in case of a breach of a security goal. Damage Classes categorize damage criterions of different domains. Damage Subclasses & Damage Criteria further partition damage classes. Damage subclasses specify the type of damage of damage classes. Damage criteria represent different levels of caused damage within a damage subclass. Required Attack Potential describe the ability of an adversary needed to perform a threat. Risk Level defines the expected risk based on a damage potential and a required attack potential. Low damage potentials and high required attack potentials result in a low risk level. Risk Potential Rating is a table defining how risk potentials are combined by joining damage potentials and required attack potentials.

3.3.2. Threat Catalogue

Threat catalogues are supportive tools to perform a risk assessment. They provide an operator with a list of threats to check against. While they provide remedy to lacking knowledge of lesser experienced security analysts, they also help more experienced analysts to not overlook potential threats because their experience has them less interested in them. There are various threat catalogues for different domains and different level of detail, such as the BSI IT-Grundschutz Catalogues[22], and threat modeling mnemonics, such as Microsoft STRIDE, which provide a clear process in how to identify possible threats.

3.3.3. Control Catalogue

Similar to threat catalogues, control catalogues give a basic overview of commonly used security measurements to mitigate identified threats and support the security analysts work.

3.4. Modifications

In this section it is described how the MoRA methodology is altered in order to allow the integration of the *Microsoft STRIDE* threat model and the Microsoft Threat Modeling Tool as source for defining catalogues.

3.4.1. Threat Catalogue

STRIDE is a threat model created by Microsoft for the use in the Microsoft Threat Modeling Tool. STRIDE stands for spoofing, tampering, repudiation, information disclosure, denial of service and elevation of privilege. These six terms represent different categories of threats. Each category consists of specific threats targeting different kind of software or hardware. STRIDE is used in this thesis to complement the MoRA methodology by acting as a threat catalogue. To apply STRIDE to a *SUD*, the *SUD* must imported into the Microsoft Threat Modeling Tool. Once the architecture is modeled, the Microsoft Threat Modeling Tool automatically generates possible threats, following the STRIDE threat model, for every connection crossing a trust boundary. A set of the list of generated threats is used as a threat catalogue in the MoRA methodology.

3.4.2. Control Catalogue

Similar to threats, the Microsoft Threat Modeling Tool also offers a control catalogue. Controls are categorized in ten categories[32]: auditing and logging, authentication, authorization, communication security, configuration management, cryptography, exception management, input validation, sensitive data and session management. Similar to the threat catalogue, each category consists of a set of controls targeting different kind of software or hardware. Threats generated by the Microsoft Threat Modeling Tool include a suggestion of possible mitigations. These suggestions can be identified and used as a control catalogue in the MoRA methodology.

3.4.3. Determine Protection Needs

The MoRA methodology originally identifies security goals by trying to apply damage criteria to every element defined in the *SUD*. As the STRIDE catalogue is used, security goals can be determined by the generated threats of the Microsoft Threat Modeling Tool. A security goal for every found threat is created, instantiating the threatened security goal class. After creating security goals for the generated threats, damage criteria are assigned to each security goal in order to define the damage potential.

3.4.4. Analyze Threats

The Microsoft Threat Modeling Tool generates threats using the STRIDE threat model and a data flow diagram representing the researched system architecture. The generated threats are used as threats in the MoRA methodology. A set of the generated threats is created and imported into the risk assessment. This implies that generated threats of the same type, threatening different elements, are created once in the *Yakindu Security Analyst*. The default value of required attack potential for generated threats is set to ”Low”. Deviating values are possible and reasoned at creation.

3.4.5. Establish Controls

Generated threats by the Microsoft Threat Modeling Tool suggest possible mitigations. A set of the suggested mitigations is created and used to create controls. These controls mitigate every threat that referenced them. The default value for the required attack potential is ”High” unless reasoned otherwise at creation of the control.

4. Analysis

The chapter *Analysis* consists of the three sections *General Data Protection Regulation*, *CliniScale System Process* and *Technical Guidelines for Cryptographic Protocols*. *General Data Protection Regulation* analyses the GDPR in order to identify regulations regarding this thesis and extract requirements. *CliniScale System Process* models the process and workflows of the CliniScale project. The third section, *Technical Guidelines for Cryptographic Protocols*, defines the used technical guidelines used for cryptographic protocols.

4.1. General Data Protection Regulation

This section analyses the statutory regulations implemented by the General Data Protection Regulation (GDPR). The GDPR is a regulation on EU-law concerning data protection and privacy for all citizens of the European Union and the European Economic Area. Its goal is to give individuals more control over their personal data and simplifying the regulatory environment for private companies by unifying the legislation within the EU.

Regulatory specifications concerning a system collecting and processing sensitive personal data in the scope of this thesis are identified. This implies that regulations by the GDPR concerning other subjects, such as Designation of the data protection officer[10] or Fines/Penalties[17], are not taken into account.

Personal Data Article 4 (1) defines personal data as any information relating to an identified or identifiable natural person. An identifiable natural person is defined as an entity that can be identified by the data directly or indirectly. Directly could be data such as name, identification number or similar. Data that can lead indirectly to identification are, among others, location data, online identifiers or factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that person.[11]

Sensitive Personal Data Article 9 defines sensitive personal data as special categories of personal data. The term sensitive personal data is found in the recital 51. It is defined as personal data that is by nature sensitive in relation to fundamental rights and freedoms of a natural person. Sensitiveness of the data merits specific protection as their processing can lead to significant risks to fundamental rights. In paragraph 1 sensitive personal data is defined as data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetic data, biometric data, data concerning health or data concerning a natural person's sex life

or sexual orientation. The processing is prohibited per default and is only possible if specific rules defined in paragraph 2 apply. The only rule of interest for this thesis of the paragraph 2 for the processing of health data in the context of the CliniScale is defined in (a). Paragraph 2 (a) allows processing if the data subject has given explicit consent and no Union or Member State law prohibits lifting the prohibition of paragraph 1.[18][14]

Lawfulness of Processing Defined in article 6 of the GDPR. The article defines under which conditions the processing of personal, and sensitive, data is lawful. Paragraph 1 defines six conditions that may apply to a company: the data subject has given consent to the processing, processing is necessary for the performance of a contract the data subject is part of, processing is necessary for compliance with a legal obligation of the controller, processing is needed in order to protect vital interests of the data subject, processing is necessary to carry out a task carried out in public interest or by official authority in the controller and if processing is necessary for the purpose of the legitimate interests of a controller or third party. Only the first condition, consent by the data subject, is applicable to the CliniScale project.[15]

Conditions for Consent Article 7 and recital 32 define the conditions for consent. Consent must be given by a clear affirmative act by the data subject that is free given, specific, informed and unambiguous. It can be given in the form of a written statement, including by electronic means, or an oral statement. In order to obtain freely given consent, the data subject must have a real choice. Any element of inappropriate pressure or influence renders the consent invalid. For a consent to be specific and informed, the data subject must be informed on the controller's identity, what kind of data will be processed for what use and for which purpose. Additionally, the data subject must be informed about his right to withdraw consent at any time. The consent must be bound to one or several clearly defined purposes, which have been sufficiently explained to the data subject. Last, the consent has to be unambiguous. It requires either a statement or a clear affirmative act.[16][8]

Security of Processing The term defined in article 32 places the responsibility on the controller to implement appropriate technical and organizational measures to secure personal data.
Paragraph 1 details four measures to ensure data privacy and security. Paragraph 1 (a) advocates to pseudonymization and encryption of personal data. Paragraph 1 (b) recommends measures to ensure the ongoing confidentiality, integrity, availability and resilience of processing systems and service. Paragraph 1 (c) includes measures to restore availability and access in a timely manner in case of a physical or technical incident. Last, paragraph 1 (d) recommends regularly testing, assessing and evaluation of the effectiveness of measures that ensure data privacy and security. These recommendations shall be implemented by the controller based on the risks to rights and freedoms of the data subjects. It is to note, that the GDPR gives no recommendation on which standards, protocols or configurations to use. The only requirement is to choose measurements by evaluating the state of the art, cost of implementation

and the nature, scope, context and purpose of the processing. To determine what exactly falls under state of the art, it is recommended to rely on security standards or IT-security guidelines.
Paragraph 2 defines that the choice of security measures should be made by assessing an appropriate level of security based on the risks present to specific processing of personal data based on accidental or unlawful destruction, loss, alteration, unauthorized disclosure or access to personal data transmitted, stored or otherwise processed. This means that a controller has to implement appropriate measures depending on the risks present to a process or personal data.
Paragraph 3 recommends complying with the code of conduct, defined in article 40[12], or make use of certification, described in article 42 [13], to demonstrate compliance with the requirements set in paragraph 1.
Paragraph 4 binds the controller to implement a process or measurements to ensure that any person acting under the authority of the controller or the processor who has access to personal data does not process them except on instruction of the controller.
Using encryption as a measurement has additional benefits for the controller. In case of a data breach, authorities positively consider the use of encryption in their decision of whether and of what amount a fine is imposed.[7]

Data Protection by Design and by Default These principles are defined in article 25 of the GDPR. It defines the responsibility of the controller to implement the data protection principles privacy by design and privacy by default. These measurements could consist of data minimization, pseudonymization of personal data as soon as possible, transparency of functions and processing of personal data or enabling the data subject to monitor the processing of his data.[6]

In summary, requirements to the CliniScale project to comply with the GDPR can be defined.

1. Personal and sensitive personal data has to be secured in a manner appropriate for the risks resulting in a loss of confidentiality, integrity or availability of the data.
2. State of the art measurements have to be implemented, such as cryptography or pseudonymization.
3. In order to gather and process personal data, consent of the user is required in a way of a clear affirmative act by the data subject that is freely given, specific, informed and unambiguous.
4. Data protection should be by design and by default.

A security risk assessment following the MoRA methodology implements security measurements based on the damage potential of an element for every security goal identified. This implies that data is secured in an appropriate way depending on the risks they provoke.
Implemented measurements follow technical guidelines recommending state of the art protocols. For further information see *Technical Guidelines for Cryptographic Protocols*.

The act of obtaining consent of a user is out of the scope of this thesis. Therefore, it is assumed consent is given in a way complying the requirements of the GDPR.
By performing a security risk assessment in the early stages of development, the practice of security by design is applied from the beginning. Security by default is a property that the CliniScale team has to ensure in process of implementation.

4.2. CliniScale System Process

In this section the process and workflow of the CliniScale project is defined.
The infrastructure of the CliniScale Project is designed as a classic Client-Server model. This means it is a distributed application infrastructure with the server as a provider of services and resources and the client as service requester. In this case, two kind of clients request different services and resources from the CliniScale back end. The participating parties and their interaction are listed below:

CliniScale Environment The server application consists of two services: the back end service and the TrialConfigurator. The TrialConfigurator offers the Trial Executor the service to create and configure a trial and select possible probands to perform the trial. Information about the probands is send to the Trial Executor in the form of ProbandData. ProbandData contains a list of possible probands the Trial Executor can choose from to perform the trial on. The Trial Executor creates a trial fitting his needs and sends the TrialConfiguration back to the TrialConfigurator. The back end service offers probands the service to register for the configured trial by using the CliniScale Application for mobile phones. After registration, the back end service sends the TrialConfiguration to the mobile application. As the proband performs the trial, information containing answers to surveys and the health data from measurements is gathered. These data are bundled as HealthData and send back to the back end service. After the configured trial period is over, the back end service bundles collected data of the probands and creates a report for the Trial Executor. This TrialResults is send to the Trial Executor.

Trial Executor The Trial Executor is able to create and configure a clinical trial by using the TrialConfigurator. In the process of creating a trial, ProbandData is send from the Trial Configurator to the Trial Executor. ProbandData contains information about possible probands the Trial Executor can choose from to perform the clinical trial. When the Trial Executor finished configuring the trial, TrialConfiguration is send to the Trial Configurator.

CliniScale Application The application for mobile phones enables probands to register for trials and collect trial results. Is a proband chosen to participate in a trial, the CliniScale back end service sends the TrialConfiguration to the CliniScale Application. The proband performs the clinical trial and collects data in the process. This data is sent back to the CliniScale back end service.

Following the defined process, the following things can be defined to model the system architecture. The system consists of three parties, the CliniScale Environment, the Trial Executor and the CliniScale Application representing the probands. The CliniScale environment consists of two services, the TrialConfigurator and the CliniScale back end. The TrialConfigurator has a connection to the Trial Executor, to enable the creation of a clinical trial. The CliniScale back end service has connections to both the Trial executor, to deliver the results to the trial, and the CliniScale Application, in order to perform the clinical trial. The Figure 4.1 represents a basic model of the defined process.

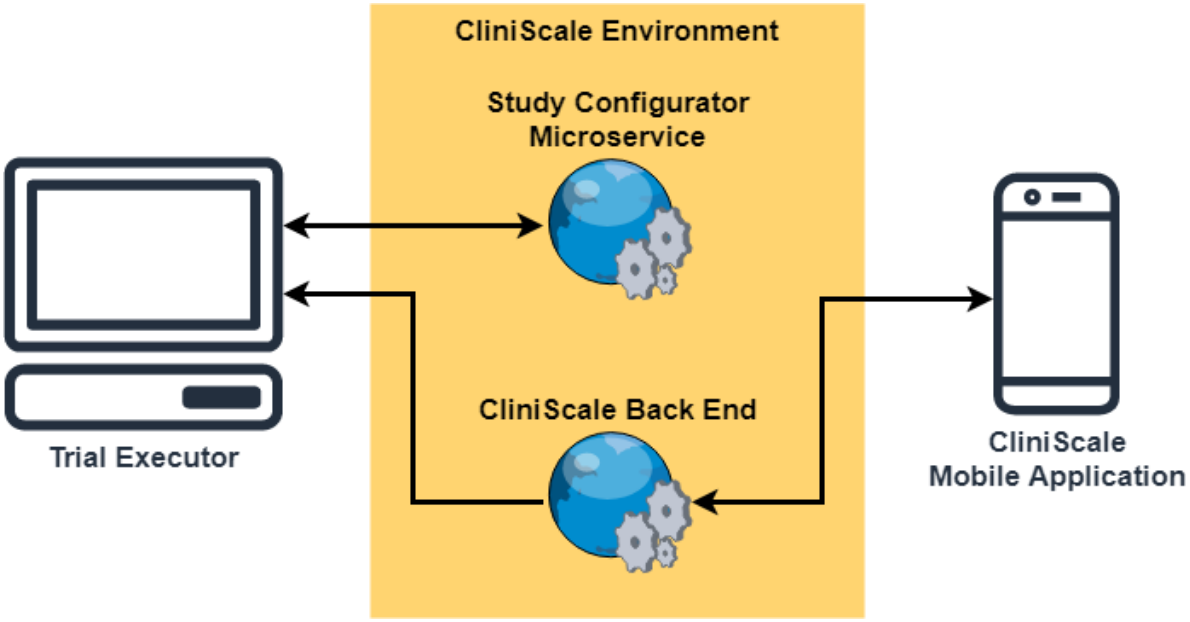

Figure 4.1.: CliniScale Process Architecture

4.3. Technical Guidelines for Cryptographic Protocols

The section *Technical Guidelines for Cryptographic Protocols* identifies a technical guideline used as source of information when choosing cryptographic protocols and standards. The GDPR specifies the use of state of the art cryptographic protocols, without specifying which protocols or standards exactly to use and how to configure them. The GDPR recommends using proven standards and IT-security guidelines.
The German Federal Office for Information Security (BSI) published technical guidelines[30] for cryptographic mechanisms assessing the security of different cryptographic protocols. From the various published guidelines, BSI TR-02102-1[1] and BSI TR-02102-2[2] are the ones relevant for this thesis.
BSI TR-02102-1 presents general recommendations for the choice of basic protocols and the right configuration of them, especially regarding the critical key length.
BSI TR-02102-2 concerns itself with the Transport Layer Security (TLS), a protocol

which is commonly used for the transportation of data in the internet.
The goal of the BSI technical guidelines is to spread appropriate security standards. Being updated regularly they provide a reliable source for information about state-of-the-art cryptographic protocols and their configuration that can be used in this thesis.

5. Architecture Design and Security Risk Assessment

This chapter describes the process of performing a security risk assessment. It consists of the three sections *Assessment Model*, *Document System Under Development* and *Security Risk Assessment*. *Assessment Model* explains the values set in the Assessment Model as preliminary to performing the risk assessment. The second section, *Document System Under Development*, explains how the CliniScale system is modeled as System Under Development and imported in the Microsoft Threat Modeling Tool. The third section, *Security Risk Assessment*, describes the performance of the security risk assessment.

In order to perform the security risk assessment, the exact scope of the area of responsibility has to be defined. As the goal of this thesis is to design a concept for the secure communication of sensitive data, the scope of the security risk assessment is narrowed down to examine the security of data and the communication of these. This implies the assumption, that within a trust boundary components and communication are secure. This risk analysis targets communication channels between components of different trust boundaries.

For identification purposes, a nomenclature is introduced to identify the basic elements of a security risk assessment. [X.i: Y] with the brackets ”[” and ”]” limiting the term. ”i” is a number to separate elements of the same type, ”Y” is the elements name. ”X” identifies the element type, being one of the following: ”F” for functions, ”Cmp” for components, ”D” for data elements, ”DF” for data flow, ”Ch” for data channels, ”TC” for threat class, ”CC” for control class, ”G” for security goal, ”T” for threat, ”R” for risk and ”C” for control.

5.1. Assessment Model

The section describes the configuration of the *Assessment Model*, configuring basic settings as a preliminary to perform a security risk assessment. The result is the creation of the artifact defined in chapter 3 section 3.3.

Security Goal Classes Security goal classes describe fundamental security properties of the elements under investigation. As the STRIDE threat model is used as a threat catalogue, the defined security properties are used as security goal classes. The following six security goal classes are created: ”Confidentiality”, ”Availability”, ”Integrity”, ”Authenticity”, ”Accountability” and ”Authorization”. Further definition of these properties can be found under in section 2.1 of chapter 2.

```
1 security goal class  Confidentiality [CON]
2 security goal class  Availability [AVA]
3 security goal class  Integrity [INT]
4 security goal class  Authenticity [AUTC]
5 security goal class  Accountability [ACC]
6 security goal class  Authorization [AUTZ]
```

Source Code 5.1: Security Goal Classes

Damage Potential To estimate the resulting damage of breaching a security goal, three damage potentials are created: ”Low”, ”Moderate” and ”High”.

```
1 damage potentials
2     Low [LOW] = 1
3     Moderate [MOD] = 2
4     High [HIG] = 3
```

Source Code 5.2: Damage Potentials

Damage Classes, Subclasses and Damage Criteria To define different categories of consequences a security goal breach can have, two damage classes are defined: ”Right to self-determination” and ”Financial consequences”.

```
1 damage class Right to self-determination [SelfDet]: Violation of
      the right to self-determination
2 damage class Financial consequences [FinCon]: Financial
     measurable damage
```

Source Code 5.3: Damage Classes

”Right to self-determination” describes violations against the name giving right to privacy of every individual, further defined in the GDPR. The damage subclass ”Loss of self-determination” defines three levels of severity within this damage class. The first, ”Leak of unlinkable and non-identifying information”, describes the loss of data that is unlinkable and cannot be used to identify any individual. Because of this, the expected damage potential ”Low” is assigned. The second criteria, ”Leak of personal data”, describes the loss of confidentiality of information that contains personal data. The expected damage potential is ”Moderate”. The third criterion is ”Leak of sensitive personal data”. It represents a situation in which sensitive personal data is leaked and the right to self-determination of an individual is critically impaired, e.g. the health records of a user are disclosed publicly. ”Leak of multiple identities” is the last damage criteria in this category. It describes the leak of data that can be used to identify multiple individuals. Both ”Leak of sensitive personal information” and ”Leak of multiple identities” have a damage potential of ”High”.

```
1 damage subclass Loss of self-determination (LossSelf) refines
   SelfDet: Right to self-determination with criteria
2     Leak of multiple identities [IdDat] = HIG: High
3     Leak of sensitive personal data [SensDat] = HIG: High
4     Leak of personal data or single identities [PerDat] =
     MOD: Moderate
5     Leak of unlinkable and non-identifying information [UnlDat]
     = LOW: Low
```

Source Code 5.4: Damage Subclass: Right to Self-determination

”Financial consequences” describes resulting monetary damages of a breach to security goals. These include fines caused by a breach of contract or violation of legal regulations such as the GDPR.
The damage subclass ”Monetary charges” covers fines due to breaches of contracts or regulations. It consists of three damage criteria. ”Negligible fine” describes potential losses without big impact on the business, such as minor contract breaches that result in a manageable fine, increased operation costs or minor lost revenue. The damage potential ”Low” is assigned to it. ”Considerable fine” defines financial losses which have a significant impact on the business but are not irrecoverable. ”Considerable fine” has the damage potential ”Moderate”. The third damage criterion is ”Existence-threatening fine”, describing financial consequences that endanger the existence of the whole business. These could be caused by breaches towards the GDPR resulting in fines that lead to bankruptcy. The damage potential ”High” is assigned.
”Reputational damage” is a damage subclass defining reputational damage of a company and resulting restrictions towards the business. Leakage of sensitive information negatively influences the reputation of the company and could result in a loss of thrust towards the company, which results in further financial damage due to the loss of customers and users. ”Negligible reputational damage” is a damage criterion describing manageable damage towards the company’s reputation. The damage potential ”Low” is assigned to the criterion. The second criterion is ”Considerable reputational damage” describing an extensive loss of reputation resulting in considerably lower user numbers and financial consequences. Therefore, the damage potential ”Moderate” is assigned. ”Existence-threatening reputational damage” defines consequences to a breach having such a high impact on the company’s reputation that the ability to perform clinical trials is impaired due to dwindling user numbers or lack of customers wanting to cooperate with the CliniScale project. Leaks of multiple identities or sensitive personal data can lead to such reputational damage. The damage potential ”High” is assigned.

```
1 damage subclass  Monetary charges (MonChar) refines
     FinCon: Financial consequences with criteria
2     Existence-threatening fine [ExcFine] = HIG: High
3     Considerable fine [ConFine] = MOD: Moderate
4     Negligible fine [NegFine] = LOW: Low
```

```
6 damage subclass  Reputational Damage (RepDmg) refines
      FinCon: Financial consequences with criteria
7      Existence-threatening reputational damage [ExcRep] =
      HIG: High
8      Considerable reputational damage [ConRep] = MOD: Moderate
9      Negligible reputational damage [NegRep] = LOW: Low
```

Source Code 5.5: Damage Subclass: Monetary Charges and Reputational Damage

Required Attack Potentials To describe the ability of the adversary needed to perform a threat, four required attack potential levels are defined: ”Low”, ”Moderate”, ”High” and ”Beyond High”. ”Beyond High” describes a scenario which makes it nearly impossible for the adversary to perform a threat, such as using state of the art cryptography.

```
1 risk levels
2      Low = 0
3      Moderate = 14
4      High = 21
5      Beyond High = 27
```

Source Code 5.6: Required Attack Potentials

Risk Levels and Risk Table To assess the risk value of risk elements, three risk levels are defined: ”Low risk”, ”Moderate risk” and ”High risk”. Further, to calculate the risk levels out of a given damage potential and a required attack potential, the risk matrix is defined as seen in the Table 5.1.

```
1 required attack potentials
2      Low risk
3      Moderate risk
4      High risk
```

Source Code 5.7: Risk Levels

Risks Table		Required attack potentials			
		Beyond High	High	Moderate	Low
Damage potentials	Low	Low risk	Low risk	Low risk	Moderate risk
	Moderate	Low risk	Low risk	Moderate risk	High risk
	High	Low risk	Moderate risk	High risk	High risk

Table 5.1.: Risk Table

5.2. Document System Under Development

This section describes how the CliniScale project is modeled as a *System Under Development (SUD)*. The applied process is defined in *Document System Under Development* and results in the work product *System Under Development*. Modeling the *SUD* is essential to perform the security risk assessment as it describes the underlying system architecture. The *SUD* is modeled after the process defined in *CliniScale System Process*.

5.2.1. Identify Functions

First step is to identify functions representing the main functionality of the CliniScale systems process. The process can be divided into three basic work steps: Configuration of the trial, performing the trial on selected probands and sending the trial results to the *Trial Executor*.

Configuration of the trial is the first process investigated. The *Trial Executor*, the party that wants to create a clinical trial, and the *Trial Configurator*, a service offered by the CliniScale project to create trials, take part in this process.
To model the first step of the process, the function ”F.1: Create a clinical trial” is created. This process is divided into two steps: The configuration of the trial and the communication of the configuration to CliniScale. To configure the trial, the *Trial Executor* obtains the data needed from the *Trial Configurator*. These include data such as forms to create surveys or lists of possible probands.
To represent this process, the sub-function ”F.1.1: Configure clinical trial” is created. Next step is to send the configured trial from the *Trial Executor* to the *Trial Configurator* service. The sub-function ”F.1.2: Send trial configuration to the Trial Configurator” models this process.

To model the execution of the trial the function ”F.2: Perform the trial” is created. This process is divided into three sub-processes: Communicating the trial to the probands mobile application, performing the trial by gathering the asked information and returning the collected information to the CliniScale back end.
”F.2.1: Send trial configuration to mobile application” represents the process of sending the trial configuration to the applications of selected probands. To model the execution of the trial the sub-function ”F.2.2: Collect asked information” is created. To represent the third sub-process, the sub-function ”F.2.3: Return collected information to the CliniScale back end” is defined.

The last step of the process, the return of collected information to the *Trial Executor*, is represented by the function ”F.3: Report trial results”. This function is divided into two sub-functions: ”F.3.1: Bundle results” and ”F.3.2: Send results to Trial Executor”.

The first sub-function describes the process of creating a result report out of the collected information of the probands. The second sub-function models the communication of this report to the *Trial Executor*.

5.2.2. Define Components

In order to define components, trust boundaries within the architecture have to be identified. Trust boundaries describe entity areas within the project where communication between components is assumed to be secure because communication never leaves the control of the operator of such trust boundary, whereas communication between components of different trust boundaries is mostly performed over the internet. This enables examination of the communication channels between components of different trust boundaries. These communication channels are the subject of research in this thesis, as the goal is to secure the communication between different parties.

First, the trust boundary representing the CliniScale environment and the components in it are defined. All components belonging to the CliniScale project itself are found within this boundary. A component called ”Cmp.1: CS Environment”, representing this trust boundary, is created. Within this component, two subcomponents representing the CliniScale back end and the *Trial Configurator* are created: ”Cmp.2: CS back end” and ”Cmp.3: CS Trial Configurator”.

The next trust boundary represents the proband using the CliniScale mobile application. The component ”Cmp.4: Mobile Phone” is defined to model the trust boundary of the mobile device. This component consists of one subcomponent representing the CliniScale application: ”Cmp.5: CS Application”.

The third trust boundary describes the *Trial Executor*. It is represented by the component ”Cmp.6: Trial Executor”. As how exactly this domain is structured is out of scope, since the *Trial Executor* is an external party, no subcomponents can be specified.

5.2.3. Define Data

Next step in modeling the *SUD* is to define the data that is collected, processed and communicated in this project. Four different kinds of data are defined in *CliniScale System Process*: HealthData, TrialConfig, TrialResults and ProbandData. For each a data element is created: ”D.1: HealthData”, ”D.2: TrialConfig”, ”D.3: TrialResults” and ”D.4: ProbandData”.

”D.1: HealthData” describes gathered data of every proband in the process of executing the trial. These data may contain medical information, making it sensitive personal data according to the GDPR. ”D.2: TrialConfig” models configuration of

the clinical trial. Besides configuration of the trial, such as survey questions, specification of the time frame or scheduling, it also contains information about selected probands to perform the trial on. This information may not contain sensitive data of the probands, but it can enable linking of data to determine information compromising a probands right to self-determination. ”D.3: TrialResults” bundles the trial results of every proband to be send to the *Trial Executor*. It contains data concerning the health of participating probands and therefore is categorized as sensitive personal data. ”D.4: ProbandData” models the information the *Trial Configurator* communicates to the *Trial Executor* to allow him to configure a clinical trial. It represents a set of probands the *Trial Executor* can choose from to perform his trial on and therefore is declared sensitive information, as the disclosure can lead to the leak of various identities.

These data elements are assigned to the components that gather, process or store them in any step of the defined process. ”D.1: HealthData” is assigned to ”Cmp.2: CS Back end”, ”Cmp.5: CliniScale Application” and ”Cmp.6: Trial Executor”. ”D.2: TrialConfiguration” is assigned to ”Cmp.2: CS Back end”. ”D.3: TrialResults” is assigned to ”Cmp.2: CS Back end” and ”Cmp.6: Trial Executor”. ”D.4: ProbandData” is assigned to ”Cmp.3: CS Trial Configurator” and ”Cmp.6: Trial Executor”.

5.2.4. Define Communication Channels

The last task in the process of modeling the *SUD* is to create channels between components to represent the communication between these. ”Ch.1: Cmp.6(Trial Executor), Cmp.3(CS Trial Configurator)” models the communication between the *Trial Executor* and the Trial Configurator. As it is a bidirectional communication, two data flows are created to represent both directions: ”DF.1: D.4(ProbandData): Cmp.3(CS Trial Configurator) -> Cmp.6(Trial Executor)” and ”DF.5: D.2(TrialConfiguration): Cmp.6(Trial Executor) -> Cmp.3(CS Trial Configurator)”. The first data flow models the communication of ”D.4: ProbandData” from the *Trial Configurator* to the *Trial Executor* to enable the configuration of the trial. The second data flow represents the communication of the data ”D.2: TrialConfiguration” from the *Trial Executor* to the *Trial Configurator*.

”Ch.2: Cmp.2(CS back end), Cmp.5(CS Application)” models the communication between the CliniScale back end and the CliniScale mobile application. Two data flows represent the functionality of the channel. The first data flow, ”DF.2: D.2(TrialConfiguration): Cmp.2(CS back end) -> Cmp.5(CS Application)”, models the transport of ”D.2: TrialConfig” from ”Cmp.2: CS back end” to ”Cmp.5: CS Application”. ”DF.3: D.1(HealthData): Cmp.5(CS Application) -> Cmp.2(CS back end)” models the other direction, transferring ”D.1: HealthData” from ”Cmp.5: Cs Application” to ”Cmp.2: CS back end”.

The last step to model the communication is to define the communication between

”Cmp.2: CS back end” and ”Cmp.6: Trial Executor” by creating the channel ”Ch.3: Cmp.2(CS back end), Cmp.6(Trial Executor)”. This channel consists of one data flow to represent the communication of ”D.3: TrialResults” from ”Cmp.2: CS back end” to ”Cmp.6: Trial Executor”: ”DF.4: D.3(TrialResults): Cmp.2(CS back end) -> Cmp.6(Trial Executor)”.

Figure System Under Development shows how the CliniScale system is modeled as *SUD*.

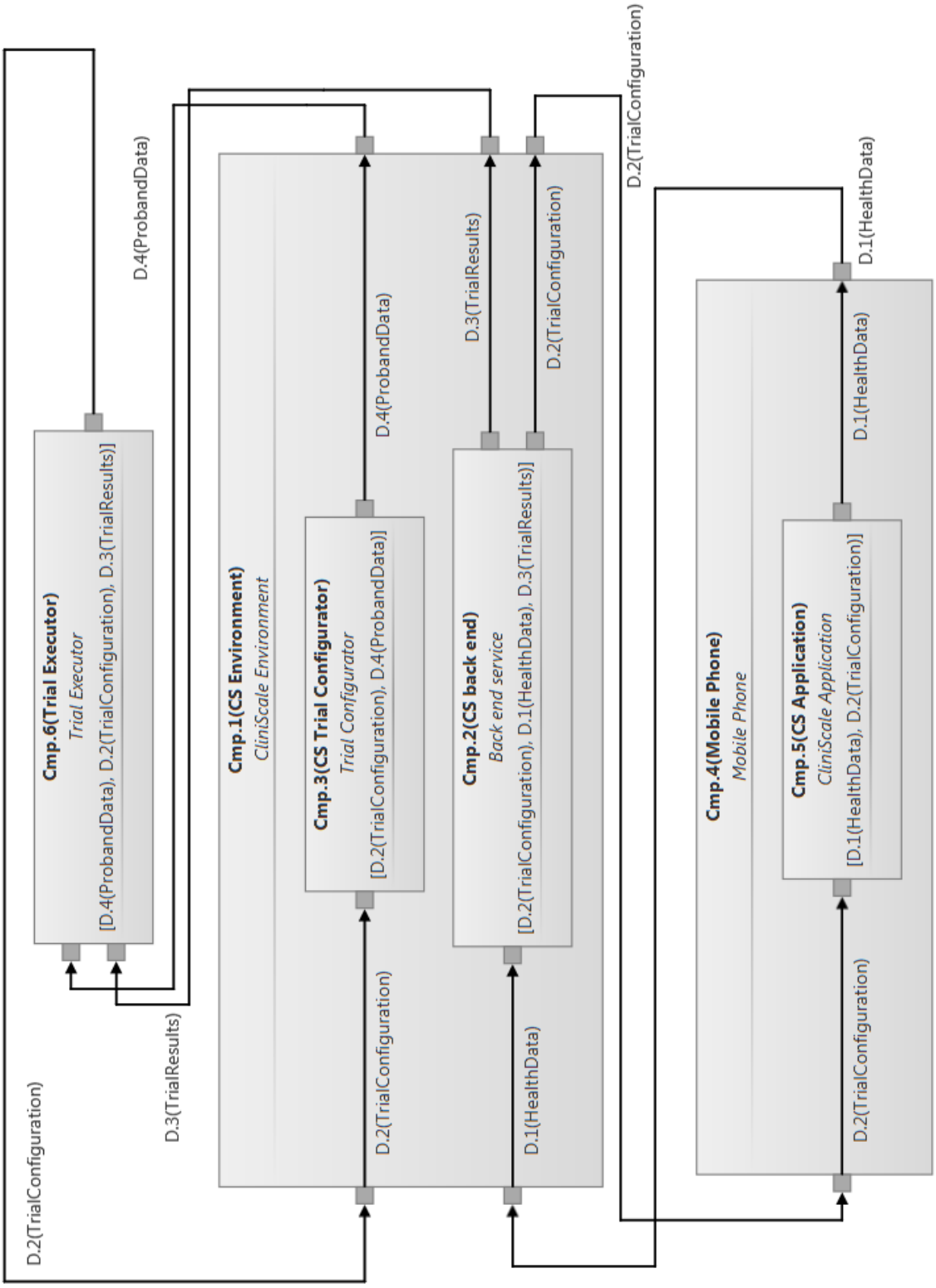

Figure 5.1.: System Under Development

5.2.5. Model the SUD in the Microsoft Threat Modeling Tool

To use the Microsoft Threat Modeling Tool as a threat and control catalogue, the *SUD* has to be modeled in it.

First, three *Generic Trust Boundaries* representing the *CliniScale Environment*, the *Trial Executor* and the mobile device running the *CliniScale application* are created. These are named *CliniScale Environment*, *Trial Executor* and *Mobile Device* representing respective components. Next, a *Web Application* and a *Web API* are created inside the *CliniScale Environment* trust boundary. The *Web Application* is named *TrialConfigurator* and the *Web API* is named *CliniScale back end* representing the components ”Cmp.3: CS Trial Configurator” and ”Cmp.2: CS back end”. To model the component ”Cmp.5: CS Application” a *Mobile Client* named *CliniScale Application* is created inside the *Mobile Device* trust boundary. The *Trial Executor* is represented as a *Browser* inside the *Trial Executor* trust boundary.

To model the channels and data flows between these components a pair of *Request* and *Response* elements is created for each data flow. The naming convention is ”DF.X_Request” for *Request* elements and ”DF.X_[Data]” for the *Response* elements containing the corresponding data. Five pairs of *Request* and *Response* elements are created in the process: ”DF.1_Request” and ”DF.1_ProbandData”, ”DF.2_Request” and ”DF.2_TrialConfig”, ”DF.3_Request” and ”DF.3_HealthData”, ”DF.4_Request” and ”DF.4_TrialResults” and ”DF.5_Request” and ”DF.5_TrialConfig”.
Figure 5.2 shows the modeling of the *SUD* in the Microsoft Threat Modeling Tool.

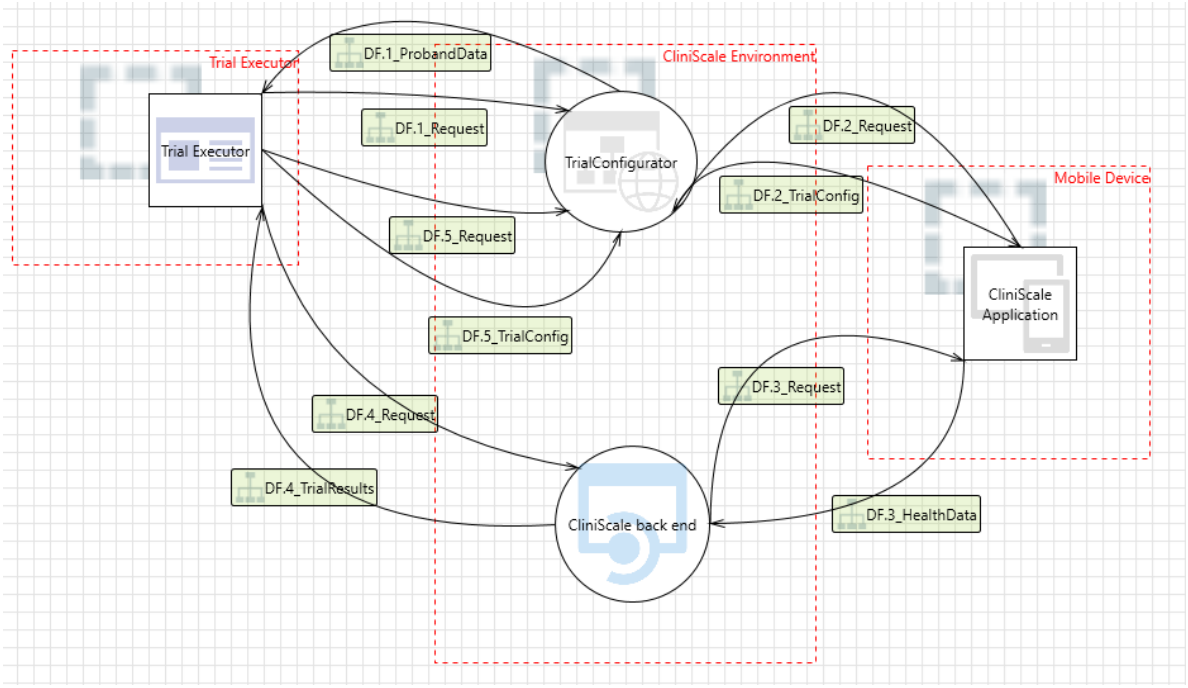

Figure 5.2.: Microsoft Threat Modeling Tool Model of the SUD

5.2.6. Generate Threat and Control Catalogues

This section describes how the Microsoft Threat Modeling Tool is used to generate threat and control catalogues applying to the *SUD*.

The Microsoft Threat Modeling Tool automatically generates threats applying the STRIDE threat model. For the created model of the *SUD*, 100 threats are generated. The generated threats contain this relevant information: name and description of the threat, STRIDE category, targeted element and possible mitigations.
The first step is to create the threat catalogue in the Yakindu Security Analyst. Six threat classes are created representing the STRIDE categories: ”TC.1: Spoofing” targeting the security goal class ”Authenticity”, ”TC.2: Tampering” targeting ”Integrity”, ”TC.3: Repudiation” targeting ”Accountability”, ”TC.4: Information disclosure” targeting ”Confidentiality”, ”TC.5: Denial of Service” targeting ”Availability” and ”TC.6: Elevation of privilege” targeting ”Authorization”. Next, a set of the exported list of generated Threats is created. The set includes 38 different threats which can be imported into the Yakindu Security Analyst as threat classes. The corresponding STRIDE category is set as refined threat class, e.g. ”TC.4.1: An adversary can gain access to certain pages or the site as a whole.” refines ”TC.4: Information disclosure”. The resulting threat catalogue consists of 44 threat classes.
The control catalogue is created in similar manner. First, control classes modeling the ten basic mitigation categories suggested by the Microsoft Threat Modeling Tool[32] are created: ”CC.1: Auditing and Logging”, ”CC.2: Authentication”, ”CC.3: Authorization”, ”CC.4: Communication security”, ”CC.5: Configuration management”, ”CC.6: Cryptography”, ”CC.7: Exception management”, ”CC.8: Input validation”, ”CC.9: Sensitive Data” and ”CC.10: Session management”.
Next step is the creation of a set of suggested mitigations by the exported list of generated threats is created. This list consists of 72 suggested mitigations. These mitigations are imported as control classes. Each control class refines the corresponding mitigation category. The resulting control catalogue consists of 82 control classes.

5.3. Security Risk Assessment

This section describes how the security risk assessment is performed following the process defined in *Methodology: Modular Risk Assessment*.

5.3.1. Determine Protection Needs

As the goal of the security risk analysis is to secure communications within the CliniScale project, the creation of security goals focuses on data and data flow elements. First step is to create a set of security goals for every data flow element. Using the

Security Goal Assessment of the Yakindu Security Analyst, a security goal for every security goal class can be generated for every data flow element. In the process 30 security goals targeting the corresponding data flow element are created.
The same procedure is repeated to create security goals for every data element, resulting in 24 more security goals. The generated security goals for data elements get assigned a depends on relation to every security goal concerning a data flow transferring matching data and matching security goal class, e.g. the security goal ”G.43: Confidentiality TrialResults” depends on ”G.19: Confidentiality D.3(TrialResults): Cmp.2(CS back end) -> Cmp.6(Trial Executor)”.
Next step is to determine damage criteria for the generated security goals. This procedure defines a damage potential for every security goal in order to estimate arising risks in case of a breach of the security properties. It creates compliance with the requirement of the GDPR to implement security measures appropriate to the risks in case of a loss of security properties. First, the damage criteria for security goals concerning data elements are set. As these security goals have a depends on relation to the security goals concerning data flows transferring the data, the damage potential defined by the assigned damage criteria is propagated towards the data flows security goals.

”D.1: HealthData” is a critical data element, as a leak of the data would breach the GDPR and have existence threatening fines and reputational damage as consequence. Therefore, the security goal concerning confidentiality, ”G.31: Confidentiality HealthData”, is assigned with the damage criteria ”Existence-threatening fine” and ”Existence-threatening reputational damage”. As sensible personal data is leaked to non-authorized parties, the damage criterion ”Leak of sensitive personal data” is also assigned. Availability of ”D.1: HealthData” induces contract breaches resulting in fines, resulting in ”Considerable fine” being assigned as damage criterion. Lacking integrity of ”D.1: HealthData” can lead to breach of contract with companies performing clinical trials with CliniScale, resulting in fines and damaged reputation. Accordingly, the damage criteria ”Considerable fine” and ”Negligible reputational damage” are assigned. Breach of authenticity can lead to a loss of confidentiality of ”D.1: HealthData”. Because of this, ”G.31: Confidentiality HealthData” is set in a depends on relation with ”G.34: Authenticity HealthData”, leading to a propagation of the damage potential towards the authenticity security goal. Accountability has a minor impact on ”D.1: HealthData” regarding contracts, therefore the damage criterion ”Negligible fine” is assigned. Lack of authorization leads to a loss of confidentiality, so ”G.36: Authorization HealthData” is added to the depends on relation of ”G.31: Confidentiality HealthData” inheriting its damage potential.

The data element ”D.2: TrialConfiguration” is critical to perform a clinical trial.
Breaches to its security goals can lead to substantial fines and reputational damage. As ”D.2: TrialConfiguration” also contains identities of possible probands, there is a risk of leaking various identities, having a grave impact on the reputation of the CliniScale project. Breach of confidentiality leads to these consequences. Therefore, the damage criteria ”Considerable fine”, ”Existence-threatening reputational damage”

and ”Leak of multiple identities” are assigned to the security goal ”G.37: Confidentiality TrialConfiguration”. Lacking availability leads to breaches of contract and minor reputational damage, as customers might want to choose a competitor when performing their next clinical trial. ”G.38 Availability TrialConfiguration” is assigned with the damage criteria ”Negligible fine” and ”Negligible reputational damage”. Integrity of the data is essential to perform a clinical trial and a breach can lead to substantial contract breaches. Therefore, the damage criteria ”Considerable fine” and ”Negligible reputational damage” are assigned. Breaches of authenticity and authorization lead to a loss of confidentiality. Both ”G.40: Authenticity TrialConfiguration” and ”G.42: Authorization TrialConfiguration” are added to the depends on relation of ”G.37: Confidentiality TrialConfiguration”. Lack of accountability may lead to minor fines due to contract breaches, so the damage criterion ”Negligible fine” is assigned to ”G.41: Accountability TrialConfiguration”.

”D.3: TrialResults” contains sensitive personal data and a breach can lead to a leak of multiple identities. Fines by the GDPR and with it, major reputational damage, are possible. Therefore, ”G.43: Confidentiality TrialResults” is a critical security goal assigned with the damage criteria ”Existence-threatening fine”, ”Existence-threatening reputational damage”, ”Leak of sensitive personal data” and ”Leak of multiple identities”. Availability of the data is important to keep up with signed contracts. ”Considerable fine” and ”Considerable reputational damage” are assigned to ”G.44: Availability TrialResults” as consequence. A breach of integrity can lead to falsified results of the clinical trial resulting in major contract breaches. Therefore ”Considerable fine” and ”Considerable reputational damage” are assigned to ”G.45: Integrity TrialResults”. Authorization and authenticity of the investigated element lead to a loss of confidentiality, therefore both security goals are added to the depends on relation of ”G.43: Confidentiality TrialResults”. Breach of accountability can lead to contract breaches resulting in considerable fines. ”Considerable fine” is assigned as damage criterion to ”G.47: Accountability TrialResults”.

The data element ”D.4: ProbandData” consists of various identities of possible probands. A breach to its security goals leads to fines defined in the GDPR, leak of multiple identities and major reputational damage. Confidentiality is a critical security property. The damage criteria ”Considerable fine”, ”Considerable reputational damage” and ”Leak of multiple identities” are assigned to ”G.49: Confidentiality ProbandData”. Lack of availability of ”D.4: ProbandData” can lead to a delay of the trial and minor contract breaches due to missing service standards. Therefore, ”Negligible fine” is assigned to ”G.50: Availability ProbandData”. Missing integrity of the data can lead to probands being chosen for a study without being qualified. This can lead to considerable contract breaches and some reputational damage. The damage criteria ”Considerable fine” and ”Negligible reputational damage” are assigned to ”G.51: Integrity ProbandData”. Lack of authorization and authenticity leads to a breach towards confidentiality, therefore ”G.52: Authenticity ProbandData” and ”G.54: Authorization ProbandData” are assigned to ”G.49: Confidentiality ProbandData” in a depends on relation. A breach of accountability can lead to minor contract breaches.

”Negligible fine” is added to ”G.53: Accountability ProbandData”

Security goals for data flows are investigated next. As the security goals for the transported data are already defined, every security goal concerning a data flow is set in a depends on relation with the matching security goal for the data element and security goal class, e.g. the security goal concerning confidentiality of ”DF.1: D.4(ProbandData): Cmp.3(CS Trial Configurator) -> Cmp.6(Trial Executor)”, ”G.1: Confidentiality D.4(ProbandData): Cmp.3(CS Trial Configurator) -> Cmp.6(Trial Executor)” is assigned into the depends on relation of ”G.49: Confidentiality ProbandData”.

In the process, 54 security goals have been created, 30 security goals concerning data flow elements and 24 security goals concerning data elements. 27 of the security goals have a damage potential ”High”, 15 a damage potential of ”Moderate” and twelve the damage potential ”Low”.

5.3.2. Analyze Threats

A threat is created for every threat class, except the threat classes modeling the STRIDE categories. In the process, 38 threats are created.
To identify the security goals every threat threatens, the exported list of generated threats by the Microsoft Threat Modeling Tool is searched for appearances of every threat to identify the threatened architecture element. By matching the STRIDE category, the corresponding security goal is found.

First, threats of the STRIDE category ”Spoofing” are investigated. The first threat ”T.1: Create fake website to phish” instantiates the threat class ”TC.1.1”. The identified security goals threatened are ”G.4”, ”G.10” and ”G.28”. ”T.2: Gain access to API end points due to unrestricted cross domain requests” instantiates ”TC.1.2” and threatens the security goal ”G.22”. The third threat, ”T.3: Gain access to user’s session due to improper logout”, instantiates ”TC.1.3” and threatens ”G.4” and ”G.28”. The threats ”T.4: Spoof targets Web Application due to insecure TLS certificate configuration” and ”T.5: Steal sensitive data like user credentials” both threaten ”G.4”, ”G.10” and ”G.28”. ”T.4” instantiates ”TC.1.4” and ”T.5” ”TC.1.5”. ”T.6: Spoof browser and gain access to Web API” instantiates ”TC.1.6” and threatens ”G.22”. The threat ”T.7: Obtain refresh/access tokens and gain access to API” instantiating ”TC.1.7” threatens ”G.16”. The threats ”T.8: Steal session cookies due to bad attributes”, instantiating ”TC.1.8”, ”T.9: Gain access to a user’s session due to insecure coding practices”, instantiating ”TC.1.9”, and ”T.10: Spoof browser to gain access to Web Application”, instantiating ”TC.1.10”, all threaten the security goals ”G.4” and ”G.28”. ”T.11: Spoof Mobile client to gain access to Web API” threatens ”G.16” and instantiates ”TC.1.11”. The last threat of the category ”Spoofing” is ”T.12: Spoof Mobile client to gain access to Web Application” and instantiates ”TC.1.12”. It threatens ”G.10”.

Next step is to create threats of the category ”Tampering”. ”T.13: Deface target web application by injecting malicious code or uploading dangerous files” instantiates ”TC.2.1” and targets ”G.3” and ”G.27”. The threat ”T.14: SQL injection through Web API” instantiates ”TC.2.2” and threatens ”G.15” and ”G.21”. ”T.15: Reverse engineer and tamper binaries” targets ”G.9” and ”G.15” and is an instance of ”TC.2.3”. ”TC.2.4” is instantiated by ”T.16: Inject malicious input into API and affect downstream processes”. The threat targets ”G.15” and ”G.21”. ”T.17: Replay attacks” instantiates ”TC.2.5” and targets ”G.9” and ”G.27”. The threats ”T.18: SQL injection through Web API” and ”T.19: Access sensitive data stored in Web App’s config files” instantiate ”TC.2.6” and ”TC.2.7” respectively and target ”G.2”, ”G.9” and ”G.27”.

Following presents threats of the category ”Repudiation”. ”T.20: Deny malicious act on API” instantiates ”TC.3.1” and threatens ”G.17” and ”G.23”. The threat ”T21: Remove attack foot prints” targets ”G.5”, ”G.11” and ”G.29”. It instantiates the threat class ”TC.3.2”.

Next, threats representing the ”Information disclosure” category of STRIDE are created. ”T.22: Gain access to certain pages or hole site” instantiates ”TC.4.1” and threatens ”G.1” and ”G.25”. The threat ”T.23: Sniff traffic from Mobile client” instantiates ”TC.4.2” and targets ”G.7” and ”G.13”. ”T.24: Access sensitive data stored in config files” is an instance of ”TC.4.3” and threatens ”G.13” and ”G.19”. ”T.25: Access sensitive information through error messages” instantiating ”TC.4.4”, ”T.27: Reverse weakly encrypted/hashed content” instantiating ”TC.4.6” and the instance of ”TC.4.7”, ”T.28: Access sensitive data from log files” all threaten ”G.1”, ”G.7” and ”G.25”. ”T.26: Gain sensitive data from mobile device” instantiates ”TC.4.5” and threatens ”G.7” and ”G.13”. The threat ”T.29: Access sensitive data from uncleared browser cache” is an instance of ”TC.4.8” and targets ”G.1” and ”G.25”. ”T.30: Access unmasked sensitive data” is an instance of ”TC.4.9” and threatens ”G.1” and ”G.25”. The threat ”T.31: Retrieve sensitive data from browser storage” instantiates ”TC.4.10” and targets ”G.19”. Both threats ”T.32: Sniff traffic to the Web API”, instance of ”TC.4.11”, and ”T.34: Access sensitive information through error messages from API”, instantiating ”TC.4.13” threaten ”G.13” and ”G.19”. ”T.33: Sniff traffic to Web Application” is an instance of ”TC.4.12” and threatens ”G.2” and ”G.25”.

”Denial of service” is the next category of created threats. The only threat of this category, ”T.35: Lack of controls against cross domain requests”, is an instance of ”TC.5.1” and threatens ”G.2” and ”G.26”.

Next are threats of the STRIDE category ”Elevation of privileges”. ”T.36: Improper validation logic” instantiates ”TC.6.1” and threatens ”G.6” and ”G.30”. The threat ”T.37: Access Web API due to poor access control checks”, instance of ”TC.6.2”, targets ”G.18” and ”G.24”. The threat ”T.38: Jailbreak mobile phone” is an instance of ”TC.6.3” and threatens ”G.12” and ”G.18”.

5.3.3. Analyze Risks

To define risks, a risk element is created for every threat defined and the threat is assigned to it using the ”caused by” relation. Based on the defined calculation in the Assessment Model, the risk level for every risk element is computed by using the required attack potential of the assigned threat and the damage potential of threatened security goals.
In the process, 38 risk elements are defined. Figure 5.3 shows a bubble chart displaying the partition of risk levels on the defined risk elements. As the overall risk level is too high for being acceptable as a security concept, controls have to be established to diminish the effects of identified threats.

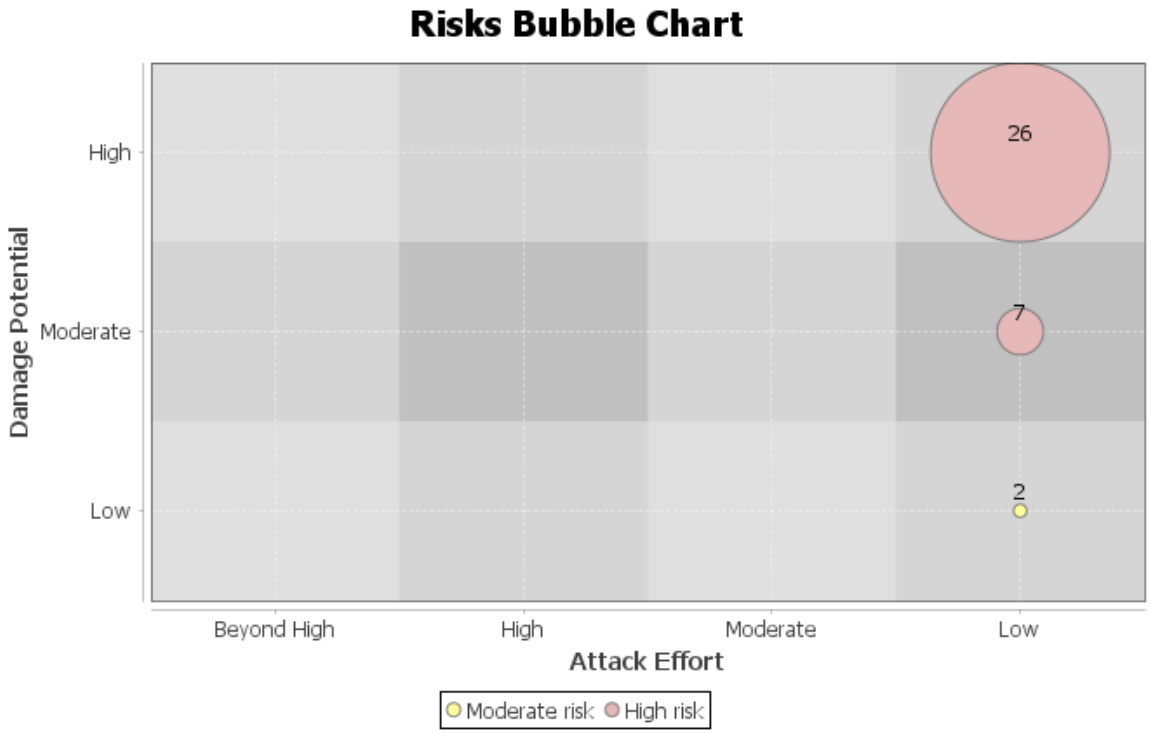

Figure 5.3.: Risk Bubble Chart

5.3.4. Establish Controls

Similar to threats, a control is created for every entry in the threat catalogue except the ten main control classes of the Microsoft Threat Modeling Tool. Mitigated threats are identified by inspecting every threat’s suggested mitigations. If it contains the current investigated control class, the threat is mitigated by it.

First, controls of the category ”Auditing and Logging” are investigated. The first control is ”C.1: Ensure that auditing and logging is enforced on Web API”, instantiating

”CC.1.1” and mitigating ”T.20”. The controls ”C.2: Ensure that log rotation and separation are in place”, instantiating ”CC.1.2”, ”C.5: Ensure that User Management Events are Logged”, instantiating ”CC.1.5”, and ”C.6: Ensure that auditing and logging is enforced on the application”, instantiating ””CC.1.6” mitigate ”T.21”. ”C.3: Ensure that the application does not log sensitive user data” mitigates ”T.28” and is an instance of ”CC.1.3”. ”T.28” and ”T.21” are mitigated by the control ”C.4: Ensure that Audit and Log Files have Restricted Access”, an instance of ”CC.1.4”. The last control of this category, ”C.7: Ensure that the system has inbuilt defences against misuse”, is an instance of ”CC.1.7” and mitigates ”T.13”.

Next, controls of the category ”Authentication” are identified. ”C.8: Consider using a standard authentication mechanism to authenticate to Web Application” is an instance of ”CC.2.1” and diminishes the threats ”T.10” and ”T.12”. The usage of proven cryptographic methods enables setting the required attack potential to ”Beyond High”. The controls ”C.9: Enable step up or adaptive authentication”, instance of ”CC.2.2”, ”C.11: Implement forgot password functionalities securely”, instance of ”CC.2.4”, and ”C.12: Ensure that password and account policy are implemented”, instance of ”CC.2.5”, all ease the effects of ”T.5”. ”C.10: Ensure that administrative interfaces are appropriately locked down” instantiates ”CC.2.3” and mitigates the effects of the threats ”T.22” and ”T.36”. The control ”C.13: Implement controls to prevent username enumeration” instantiates ”CC.2.6” and mitigates ”T.25”. ”T.6” and ”T.11” are mitigated by an instance of ”CC.2.7”, ”C.14: Ensure that standard authentication techniques are used to secure Web APIs”. As cryptographic standards can be used, providing proven security mechanisms, the required attack potential is set to ”Beyond High”. The control ”C.15: Use ADAL libraries to manage token requests from OAuth2 clients to AAD” eases the effects of ”T.7”. It is an instance of ”CC.2.8”.

The third category of controls that are investigated is ”Authorization”. ”C.16: Enforce sequential step order when processing business logic flows”, ”C.17: Ensure that proper authorization is in place and principle of least privileges is followed”, ”C.18: Business logic and resource access authorization decisions should not be based on incoming request parameters” and ”C.19: Ensure that content and resources are not enumerable or accessible via forceful browsing” all mitigate the effects of ”T.36”. They are instances of, in order, ”CC.3.1”, ”CC.3.2”, ”CC.3.3” and ”CC.3.4”. The control ”C.20: Implement implicit jailbreak or rooting detection” is an instance of ”CC.3.5”. It reduces the effects of ”T.38”. ”C.21: Implement proper authorization mechanism in ASP.NET Web API” is an instance of ”CC.3.6”. The required attack potential is set to ”Beyond High”, as modern cryptography standards are used. It mitigates the threat ”T.37”.

Next, the category ”Communication Security” is investigated. Since it applies the use of proven cryptographic standards on communication, the required attack potential of its controls is set to ”Beyond High” as default value. ”C.22: Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections” is an instance of ”CC.4.1” and mitigates the threats ”T.1”, ”T.4” and ”T.27”. The control ”C.23: Enable HTTP

Strict Transport Security (HSTS)" instantiates "CC.4.2" and mitigates the effects of "T.13" and "T.33". "C.24: Implement Certificate Pinning" instances "CC.4.3" and mitigates "T.23". Last control of the category is "C.25: Force all traffic to Web APIs over HTTPS connection". It's an instance of "CC.4.4" and diminishes the effects of "T.17" and "T.32".

"Configuration Management" is the next category proposed by the Microsoft Threat Modeling Tool that is investigated. "C.26: Implement Content Security Policy (CSP), and disable inline javascript" and "C.27: Enable browser's XSS filter" are instances of "CC.5.1" and "CC.5.2" respectively. They mitigate the threat "T.13". The control "C.28: ASP.NET applications must disable tracing and debugging prior to deployment" mitigates "T.25" and is an instance of "CC.5.3". "C.29: Ensure that only trusted origins are allowed if CORS is enabled on ASP.NET Web Applications" instances "CC.5.4" and mitigates the threats "T.2" and "T.35". "C.30: Enable ValidateRequest attribute on ASP.NET Pages" is an instance of "CC.5.5" and mitigates "T.9" and "T.13". The controls "C.31: Use locally-hosted latest versions of JavaScript libraries", instance of "CC.5.6", "C.32: Disable automatic MIME sniffing", instance of "CC.5.7", and "C.35: Access third party javascripts from trusted sources only", instance of "CC.5.10", all mitigate "T.13". "C.33: Encrypt sections of Web API's configuration files that contain sensitive data" is an instance of "CC.5.8" and mitigates "T.24". The control "C.34: Ensure authenticated ASP.NET pages incorporate UI Redressing" diminishes the effects of "T.1" and "T.35". It instantiates of "CC.5.9".

The next investigated category is "Cryptography". The required attack potential of controls of this category is set to "Beyond High", as they all propose the implementation of cryptographic standards with proven security levels. The seven controls of this category all mitigate "T.27". They define the use of cryptographic standards and their configuration. "CC.6.1" is instantiated by "C.36: Use only approved symmetric block ciphers and key lengths". "C.37: Use approved block cipher modes and initialization vectors for symmetric ciphers" instantiates "CC.6.2". "C.38: Use approved asymmetric algorithms, key lengths, and padding" is an instance of "CC.6.3". "CC.6.4" is instantiated by the control "C.39: Use approved random number generators". The control "C.40: Do not use symmetric stream ciphers" instantiates "CC.6.5". "C.41: Use approved MAC/HMAC/keyed hash algorithms" instantiates "CC.6.6". The last control of this category, "C.42: Use only approved cryptographic hash functions" is an instance of "CC.6.7".

"Exception Management" is the seventh category where controls are identified. "C.43: Ensure that proper exception handling is done in ASP.NET Web API" instantiates "CC.7.1" and mitigates "T.34". "T.25" and "T.27" are mitigated by the controls "C.44: Do not expose security details in error messages", "C.45: Implement Default error handling page" and "C.46: Set Deployment Method to Retail in IIS". They represent instances of "CC.7.2", "CC.7.3" and "CC.7.4" respectively. "C.47: Exceptions should fail safely" instantiates "CC.7.5" and helps mitigating "T.25".

The next category of controls investigated is ”Input Validation”. ”C.48: Ensure that each page that could contain user controllable content opts out of automatic MIME sniffing”, instance of ”CC.8.1”, and ”C.49: Ensure appropriate controls are in place when accepting files from users”, instance of ”CC.8.2”, both mitigate the effect of threat ”T.13”. The control ”C.50: Ensure that type-safe parameters are used in Web Application for data access” instantiates ”CC.8.3” and mitigates ”T.18”. The controls ”C.51: Encode untrusted web output prior to rendering”, ”C.53: Do not assign DOM elements to sinks that do not have inbuilt encoding” and ”C.56: Avoid using Html.Raw in Razor views” all diminish the effects of ”T.9” and ”T.13”. They instantiate ”CC.8.4”, ”CC.8.6” and ”CC.8.9” respectively. ”C.52: Perform input validation and filtering on all string type Model properties” is an instance of ”CC.8.5” and ”C.55: Implement input validation on all string type parameters accepted by Controller methods” an instance of ”CC.8.8”. Both mitigate the effects of ”T.5” and ”T.13”. ”C.54: Validate all redirects within the application are closed or done safely” is an instance of ”CC.8.7” and mitigates the threats ”T.1” and ”T.5”. ”C.57: Ensure that model validation is done on Web API methods”, instance of
”CC.8.10”, and ”C.58: Implement input validation on all string type parameters accepted by Web API methods”, instance of ”CC.8.11”, both mitigate ”T.16”. The control ”C.59: Ensure that type-safe parameters are used in Web API for data access”, instance of ”CC.8.12”, eases the effects of ”T.14”. The last control of the category is ”C.60: Sanitization should be applied on form fields that accept all characters such as rich text editor”. It’s an instance of ”CC.8.13” and mitigates ”T.9” and ”T.13”.

The ninth category of controls proposed by the Microsoft Threat Modeling Tool is ”Sensitive Data”. ”C.61: Ensure that sensitive content is not cached on the browser” is an instance of ”CC.9.1” and mitigates the threat ”T.29”. The control ”C.62: Encrypt sections of Web App’s configuration files that contain sensitive data” is an instance of ”CC.9.2”. Due to the security level of applied encryption, the required attack potential is set to ”Beyond High”. It mitigates the effect of ”T.19”. ”C.63: Explicitly disable the autocomplete HTML attribute in sensitive forms and inputs” instantiates ”CC.9.3” and mitigates ”T.5”. The control ”C.64: Ensure that sensitive data displayed on the user screen is masked” diminishes the threats of ”T.30”. It’s an instance of ”CC.9.4”. ”CC.9.5” is instantiated by ”C.65: Ensure that sensitive data relevant to Web API is not stored in browser’s storage”. This control mitigates ”T.31”. ”C.66: Encrypt sensitive or PII data written to phones local storage” is an instance of ”CC.9.6” and mitigates ”T.26”. The controls required attack potential is raised to ”Beyond High” due to the nature of enforced cryptographic methods. ”C.67: Obfuscate generated binaries before distributing to end users” mitigates ”T.15” and instantiates ”CC.9.7”.

The last category of controls is ”Session Management”. ”C.68: Applications available over HTTPS must use secure cookies” is an instance of ”CC.10.1” and diminishes the effects of ”T.8” and ”T.33”. The control ”C.69: All http based application should specify http only for cookie definition” is an instance of ”CC.10.2” and mitigates ”T.8”. ”C.70: Mitigate against Cross-Site Request Forgery (CSRF) attacks on ASP.NET web pages” mitigates the threats ”T.2” and ”T.35”. It’s an instance of ”CC.10.3”. The con-

trols ”C.71: Set up session for inactivity lifetime”, instance of ”CC.10.4”, and ”C.72: Implement proper logout from the application”, instance of ”CC.10.5”, both mitigate ”T.3”.

5.3.5. Analyze Need for Further Iterations

The effects of the controls on their mitigated threats is computed by the Yakindu Security Analyst. The risks created in *Analyze Risks* compute a new risk level for every risk as shown in Figure 5.4.

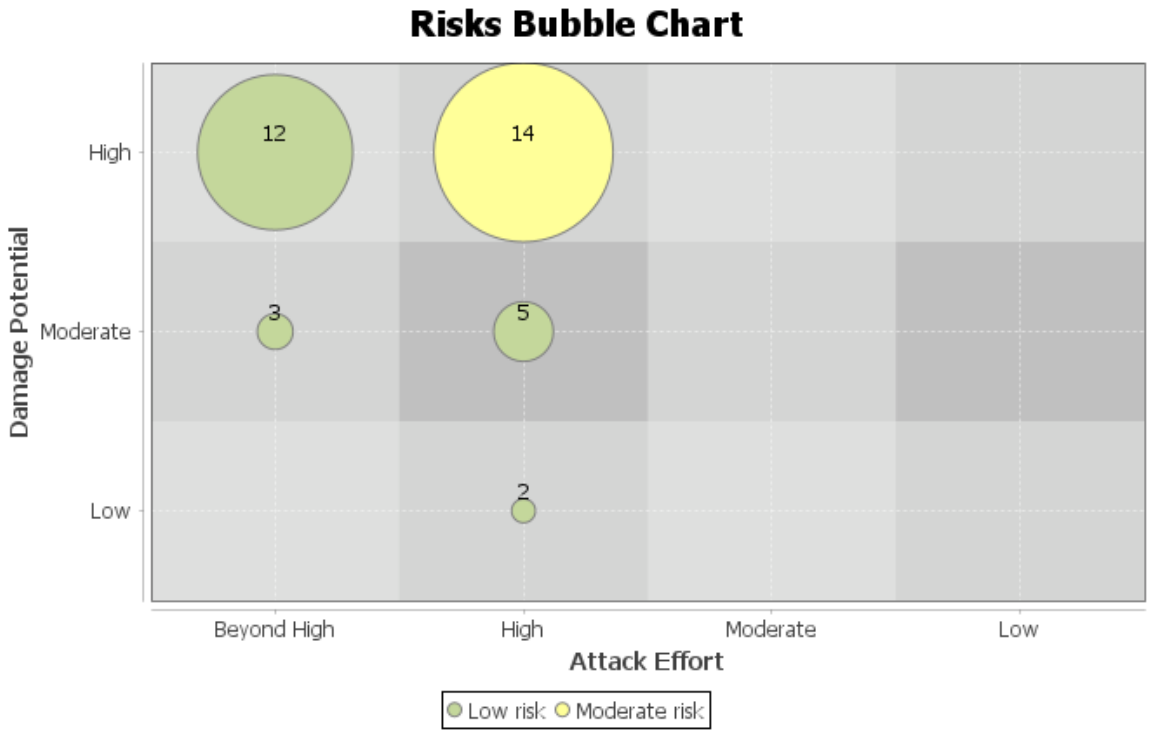

Figure 5.4.: Final Risk Bubble Chart

15 of the risks have a risk level of ”Moderate” and 23 a risk level of ”Low”. This is acceptable as an overall risk level.
Still, some of the identified controls introduce new elements to the *SUD* needed for their implementation. Specifically, controls that implement cryptographic protocols, such as ”C.25”, need a secure storage to save cryptographic keys. To model these new components, the *SUD* is altered by creating three new components: ”Cmp.7: Secure Storage (CliniScale)”, ”Cmp.8: Secure Storage (Mobile Device)” and ”Cmp.9: Secure Storage (Trial Executor)”. New data elements modeling the public and secret keys saved in each storage are defined: ”D.5: Public Key CliniScale”, ”D.6: Private Key CliniScale”, ”D.7: Public Key Trial Executor”, ”D.8: Private Key Trial Executor”, ”D.9: Public Key Mobile Device” and ”D.10: Private Key Mobile Device”.
Next, the defined data elements are assigned to the storage elements. Each storage

element contains the public and private key corresponding to his party and the public keys of every party there is a connection to. ”Cmp.7: Secure storage (CliniScale)” stores ”D.5: Public Key CliniScale”, ”D.6: Private Key CliniScale”, ”D.7: Public Key Trial Executor” and ”D.9: Public Key Mobile Device”. ”Cmp.8: Secure Storage (Mobile Device)” stores ”D.5: Public Key CliniScale”, ”D.9: Public Key Mobile Device” and ”D.10: Private Key Mobile Device”. ”Cmp.9: Secure Storage (Trial Executor)” stores ”D.5: Public Key CliniScale”, ”D.7: Public Key Trial Executor” and ”D.8: Private Key Trial Executor”.

Figure 5.5 shows the updated *SUD*. As the scope of this thesis is securing the transfer of data and there is no new data flow introduced, the security risk assessment is therefore completed at this point.

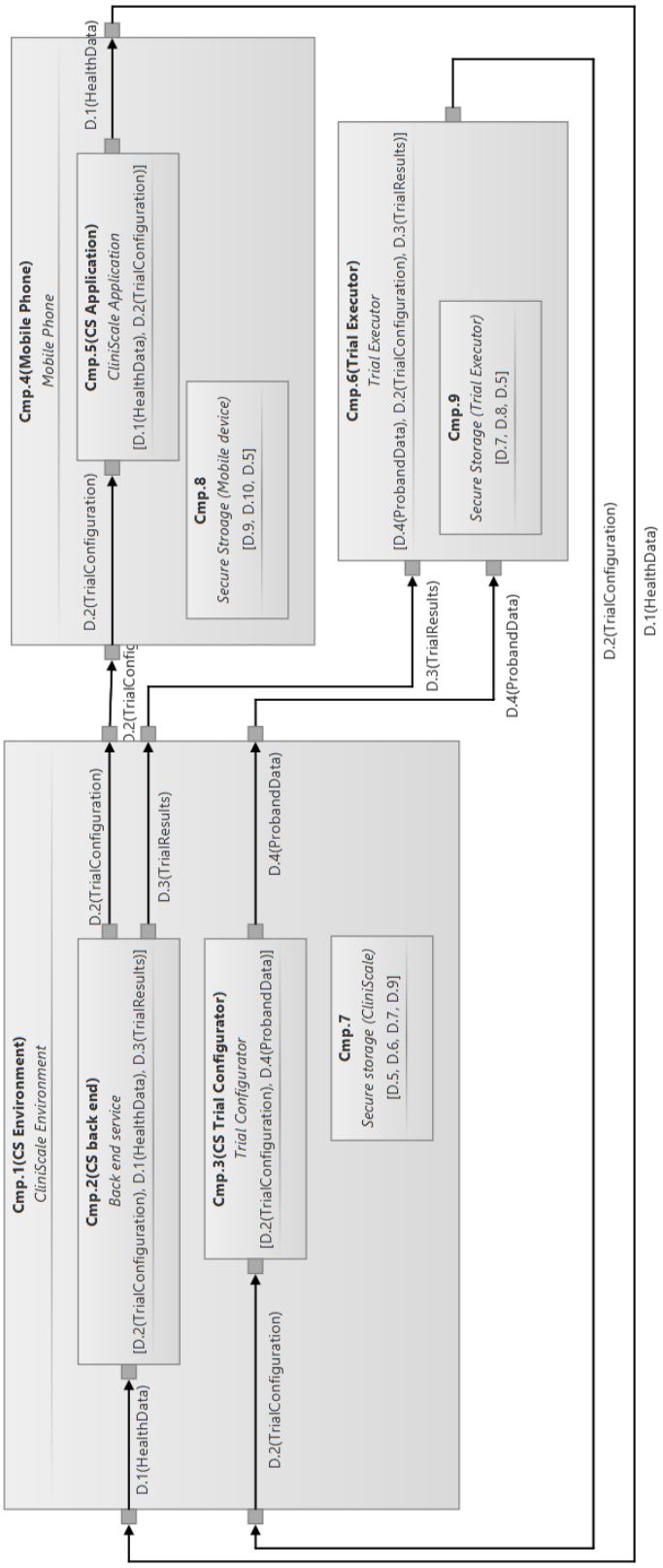

Figure 5.5.: Updated SUD

6. Implementation

The chapter *Implementation* consists of the three sections *CliniScale System Architecture*, *Technologies* and *Consequences for the Implementation*. *CliniScale System Architecture* defines the architecture of the current implementation of the CliniScale back end and mobile application. *Technologies* presents the frameworks and libraries used in the implementation. The third section, *Consequences for the Implementation*, proposes guidance in how to implement identified controls.

6.1. CliniScale System Architecture

This section presents the system architecture of the current implementation of the CliniScale project, both the back end services and the mobile application.
The back end consists of the microservices *Gateway* and *Study Configurator*.
The *Gateway*, implemented using the Spring Netflix Zuul framework, provides routing and load balance functionality. It routes incoming *REST* requests to the *Study Configurator* microservice. It also implements basic user authentication by using *JSON Web Token*.
The *Study Configurator* microservice reacts to incoming *REST* requests by gathering and processing the asked information in case of a successful authentication.
Both the gateway and the *Study Configurator* are connected to a *MySQL*[1] database, where information is stored and retrieved on demand.

The Android mobile application is coded in *Kotlin*, the official supported programming language for development for Android[2].

Figure 6.1 presents a model of the CliniScale system architecture.

6.2. Technologies

The section *Technologies* presents the frameworks and libraries used in the implementation of the CliniScale back end and the CliniScale mobile application. Following the frameworks and libraries relevant to this thesis are presented:

[1] MySQL: https://www.mysql.com/. (Online; last accessed: November 18, 2019)
[2] Android: https://www.android.com/. (Online; last accessed: November 18, 2019)

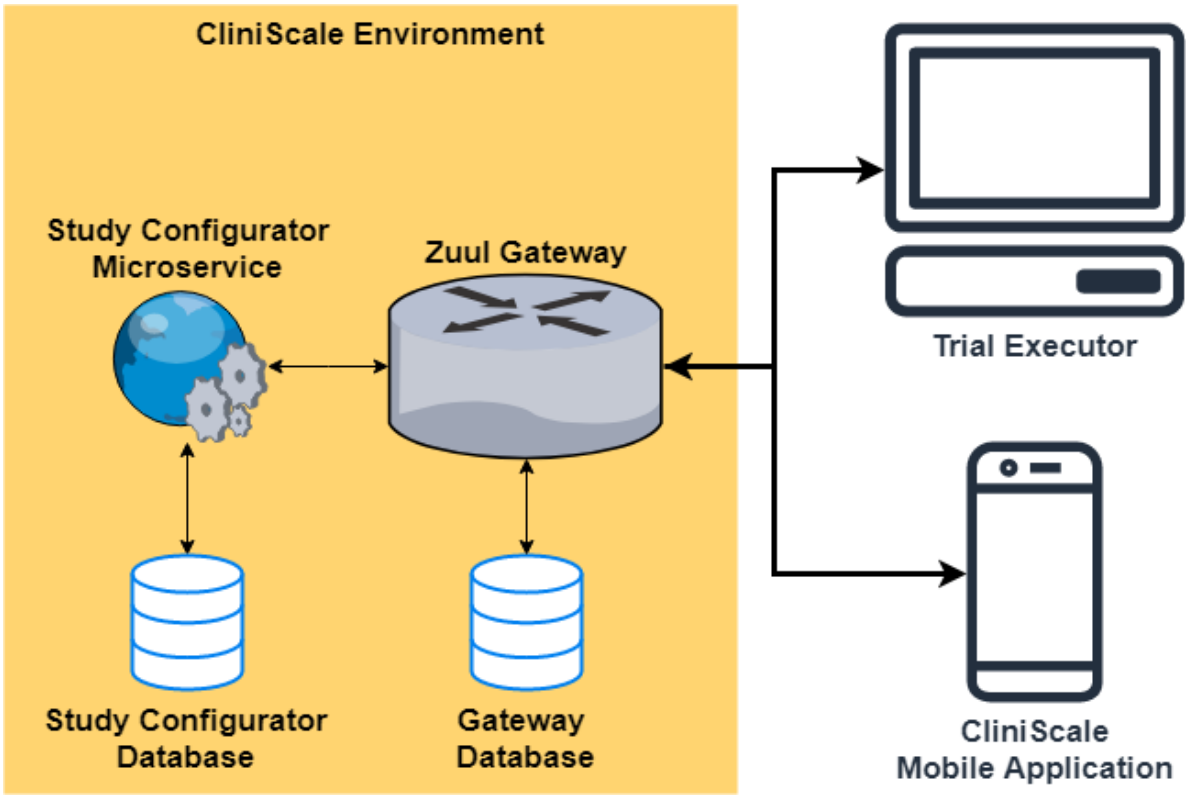

Figure 6.1.: CliniScale System Architecture

Spring Security [3] Spring security is a framework that focuses on offering highly customizable authentication, authorization and other security features to Java[4] applications. It supports a wide range of security features and functions and offers a high customizability.

Logback [5] This logging framework was created as a successor to the *log4j*[6] project. *Logback* improves the implementation of *log4j* with a number of improvements, such as faster implementation, extended tests or an improved documentation.

Zuul [7] Spring Zuul is a JVM-based router and server-side load balancer developed by Netflix. It is used as gateway in the CliniScale back end. It routes incoming requests to the appropriate micro-services. The Spring integration is part of the Spring Cloud Netflix framework[8]. Among the features, the following are of interest for this thesis:

[3] Spring Security: https://spring.io/projects/spring-security. (Online; last accessed: November 18, 2019)
[4] Java: https://www.java.com/en/. (Online; last accessed: November 18, 2019)
[5] Logback: http://logback.qos.ch/. (Online; last accessed: November 18, 2019)
[6] Log4j: https://logging.apache.org/log4j/2.x/. (Online; last accessed: November 18, 2019)
[7] Spring Zuul Documentation: https://cloud.spring.io/spring-cloud-netflix/multi/multi__router_and_filter_zuul.html. (Online; last accessed: November 18, 2019)
[8] Spring Cloud Netflix: https://spring.io/projects/spring-cloud-netflix. (Online; last accessed: November 18, 2019)

authentication and security. Zuul supports the use of the OAuth2 protocol[9], which is an industry standard authentication protocol.

Spring Boot [10] This is a framework used to create microservices in Java. It provides the user with many features to build standalone and production ready spring applications.

Kotlin [11] Kotlin is a programming language introduced by JetBrains[12]. Kotlin is designed to fully interoperate with the Java[13] and is compatible with all standard libraries of Android.

6.3. Consequences for the Implementation

This section defines basic implementation and configuration principles to implement the identified controls of the security risk assessment into the system architecture of the CliniScale project. Its subsections represent the ten categories used by the Microsoft Threat Modeling Tool to categorize controls. In each category, basic principles are defined suggesting on how to best implement controls identified in the process of the security risk assessment and which part of the CliniScale system architecture are concerned. Microsoft suggests prioritizing the categories ”Input Validation”, ”Authentication” and ”Authorization”.

6.3.1. Auditing and Logging

Auditing and logging present a set of security controls mainly concerned with mitigating threats targeting the security property of ”Accountability”.
In the process of the security risk assessment, seven control classes have been identified in this category.

- CC.1.1: Ensure that auditing and logging is enforced on Web API
- CC.1.2: Ensure that log rotation and separation are in place
- CC.1.3: Ensure that the application does not log sensitive user data
- CC.1.4: Ensure that Audit and Log Files have Restricted Access
- CC.1.5: Ensure that User Management Events are Logged
- CC.1.6: Ensure that auditing and logging is enforced on the application

[9] OAuth2: https://oauth.net/2/. (Online; last accessed: November 18, 2019)
[10] Spring Boot: https://spring.io/projects/spring-boot. (Online; last accessed: November 18, 2019)
[11] Kotlin: https://kotlinlang.org/. (Online; last accessed: November 18, 2019)
[12] JetBrains: https://www.jetbrains.com/. (Online; last accessed: November 18, 2019)
[13] Android: https://www.android.com/. (Online; last accessed: November 18, 2019)

- CC.1.7: Ensure that the system has inbuilt defences against misuse

Controls of this category affect both the CliniScale environment and the mobile application. These controls can be divided into two areas of application: Auditing/Logging management and Auditing/Logging usage.

Auditing/Logging Management This category consists of controls regarding the implementation, configuration and management of auditing and logging mechanisms. The control classes ”CC.1.1”, ”CC.1.2”, ”CC.1.4” and ”CC.1.6” are meant with it. ”CC.1.1” and ”CC.1.6” advocate to implement auditing and logging in the application, be it the back end micro-services or the mobile application. ”CC.1.2” recommends using two processes: log rotation and log separation. Log rotation describes the process of automatically naming, archiving and compressing logs based on different metrics such as time intervals or size of the log. Usually the goal is to create an automated system that archives logs in a set time interval to ease log management. *Logback*, the logging framework used in the CliniScale system, supports log rotation offering the configuration of *Appenders*[14]. Log separation describes the process of sorting log files on a different partition as where the OS/application is running in order to avert a Denial of service attack. *Logback* supports the configuration of the folder in which logs get stored in the process of log rotation. ”CC.1.4” advocates to limit access to stored log files. It is recommended that applications have write-only access and operators and support personnel have read-only access. Only administrators should have full access to the log files.

Auditing/Logging Usage Controls of this category describe principles on what to, or specifically not to, log. The control classes ”CC.1.3”, ”CC.1.5” and ”CC.1.7” advocate on which events should, or should not, be logged. ”CC.1.3” advocates to not log sensitive data, like user credentials, health information or encryption keys. Logging these kind of data opens additional ways for an adversary to obtain the data. ”CC.1.5” proposes to log all user management events, such as successful and failed logins, password resets or user registrations. This enables to detect and react to suspicious behavior. ”CC.1.7” advises logging security exceptions to enable detection of suspicious activities. It is recommended to have security exceptions for issues such as input validation, file upload violations or brute forcing of user logins. If such an exception occurs, further measurements, depending on the exception, should prevent further abuse. If an adversary tries to brute force user credentials, the account should be suspended for a period of time and a notification to the user should be send.

[14] Logback Appenders: http://logback.qos.ch/manual/appenders.html. (Online; last accessed: November 18, 2019)

6.3.2. Authentication

Proving an entities identity is crucial for the security of an application. Controls of this control class ensure the integrity of the name giving security goal.
This category consists of eight control classes:

- CC.2.1: Consider using a standard authentication mechanism to authenticate to Web Application
- CC.2.2: Enable step up or adaptive authentication
- CC.2.3: Ensure that administrative interfaces are appropriately locked down
- CC.2.4: Implement forgot password functionalities securely
- CC.2.5: Ensure that password and account policy are implemented
- CC.2.6: Implement controls to prevent username enumeration
- CC.2.7: Ensure that standard authentication techniques are used to secure Web APIs
- CC.2.8: Use ADAL libraries to manage token requests from OAuth2 clients to AAD (or on-premises AD)

The controls mainly concern the back end of the CliniScale project. ”CC.2.1” and ”CC.2.7” recommend using standard authentication mechanisms, such as credentials consisting of username and password, to access CliniScale services. ”CC.2.2” advocates the use of mechanisms such as two-factor authentication or the usage of One-Time-Passwords send via mail or SMS to further authenticate an user when logging in, accessing sensitive information or making critical changes to an account such as changing the password. ”CC.2.3” requires administrative interfaces to be secured in an advanced manner by requiring two-factor authentication or setting it up to only be accessible from specific IP ranges. ”CC.2.4” and ”CC.2.5” advocate on password policies and the handling of password retrieval. Per default, a strong password policy should be implemented to prevent dictionary based or brute force attacks. Measures should be implemented to force users to create complex passwords, such as minimum password length, usage of special characters and the use of two-factor authentication. A secure password policy should also implement measures to protect user accounts from adversaries. A soft lock-out, disabling the account for a set time-frame after repeatedly entering the wrong password, protects against brute force attacks on user accounts. A hard lock-out can help if an account is performing malicious activities. Additionally, verify that standard passwords of software or hardware are changed after installation. To generate passwords, a password generator based on a proven random number generator should be used. Password recovery should be implemented using two-factor authentication whenever possible. To implement the control class ”CC.2.6” it is required to set up a generalized error message to prevent adversaries from obtaining additional information. ”CC.2.8” is a Microsoft specific control class that does not apply to the CliniScale project.

6.3.3. Authorization

Authorization ensures that services and data is only accessed by parties with the rights to do so. Six control classes have been identified in this category:

- CC.3.1: Enforce sequential step order when processing business logic flows
- CC.3.2: Ensure that proper authorization is in place and principle of least privileges is followed
- CC.3.3: Business logic and resource access authorization decisions should not be based on incoming request parameters
- CC.3.4: Ensure that content and resources are not enumerable or accessible via forceful browsing
- CC.3.5: Implement implicit jailbreak or rooting detection
- CC.3.6: Implement proper authorization mechanism in ASP.NET Web API

These controls concern both the back end and the mobile application. ”CC.3.1” advocates to implement mechanism to prevent automatic exploitation of offered services, for example by running a bot, by an adversary to gather as much information as possible. These mechanisms could be logic that checks for processes to be run in sequential order, without skipping single steps or passing steps in a time that no genuine human being could. ”CC.3.4” goes hand in hand with this, suggesting implementing logic to prevent acquiring large amounts of data by enumerating requests, e.g. by implementing CAPTCHA controls or randomized identifiers. ”CC.3.2” advocates to follow the principle of least privileges. This principle requires users to have no rights at all per default. Specific user roles get exactly the privileges they need so the user can perform the work he is intended to be able to perform. A user whose job is to manage the database does not need the rights to install new software. Same goes for processes, if a normal user does not need access to specific data, he should not have the right to do it. Using the Spring Security framework, user roles and privileges can be defined to implement this principle. ”CC.3.3” advocates to perform all user access restriction checks server-side. To do this, the UserID could be stored in the session cookie and checked on server-side whenever a user wants to access data or services. The control class ”CC.3.5” recommends implementing jailbreak or rooting detection to safeguard the applications configuration. These mechanisms should be implemented to check on startup of the application and deny execution of the application in case of a jailbreaked or rooted mobile device. ”CC.3.6” advocates to implement authorization mechanism in the Web API. Spring Security offers solutions to this problem by supporting roles and privileges which can be assigned to user classes.

6.3.4. Communication Security

The category ”Communication Security” ensures all communication is done in a as secure as possible manner. The four control classes recommend using proven secure communication standards in order to ensure the security and privacy of the communication channels.

- CC.4.1: Verify X.509 certificates used to authenticate SSL, TLS, and DTLS connections
- CC.4.2: Enable HTTP Strict Transport Security (HSTS)
- CC.4.3: Implement Certificate Pinning
- CC.4.4: Force all traffic to Web APIs over HTTPS connection

Implementation of these control classes concerns the CliniScale back end. ”CC.4.2” and ”CC.4.4” recommend using the Hypertext Transfer Protocol Secure (HTTPS) protocol for every connection. The HTTPS protocol is an extension of the HTTP protocol, encrypting it using Transport Layer Security (TLS), or its predecessor, Secure Socket Layer (SSL). Using HTTPS offers authentication of the website and integrity and privacy of the content of the communication. When using two-way TLS, or TLS with client authentication, a client certificate is also involved hardening the authentication process. By enabling HSTS, all communication to and from the server is forced to use HTTPS. HTTPS offers protection against Man-in-the-middle attacks, eavesdropping and tampering of the communication. Spring Security supports the use of most basic TLS versions. The BSI TR-02102-2 suggests using the TLS version 1.2 or 1.3 with enabled Perfect Forward Security (PFS). PFS is a feature protecting past sessions against future compromise of secret keys by generating a unique session key for every session. Any cipher-suite and configuration recommended by the TR-02102-2 can be used to ensure a security level that holds up till at least the year 2025. ”CC.4.1” advocates to verify the X.509 certificate of an SSL, TLS or DTLS connection. X.509 is a public key infrastructure standard that can be used to verify the identity of a user and that a public key is contained within a certificate signed by a trusted certificate authority. These certificates are routinely used to verify the identity of an entity within the internet. In the process of verifying a certificate, many attributes can be audited, e.g. domain name, trust chains or key lengths. In order to provide a high level of trust, as many attributes as possible should be verified. ”CC.4.3” advises to implement certificate pinning in a mobile application. Certificate pinning describes the process of associating a host with their expected X.509 certificate or public key. If an attacker tries to start a Man-in-the-middle attack, the certificate or public key would differ from the saved one.

6.3.5. Configuration Management

Configuration management refers to the management of configurations for an information system with the goal of enabling security managing risk. It consists of ten control classes:

- CC.5.1: Implement Content Security Policy (CSP), and disable inline javascript
- CC.5.2: Enable browser's XSS filter
- CC.5.3: ASP.NET applications must disable tracing and debugging prior to deployment
- CC.5.4: Ensure that only trusted origins are allowed if CORS is enabled on ASP.NET Web Applications
- CC.5.5: Enable ValidateRequest attribute on ASP.NET Pages
- CC.5.6: Use locally-hosted latest versions of JavaScript libraries
- CC.5.7: Disable automatic MIME sniffing
- CC.5.8: Encrypt sections of Web API's configuration files that contain sensitive data
- CC.5.9: Ensure that authenticated ASP.NET pages incorporate UI Redressing or click-jacking defenses
- CC.5.10: Access third party javascripts from trusted sources only

These control classes are implemented in the back end. "CC.5.1" suggests implementing CSP to protect against different sort of attacks, such as XSS, data exfiltration or click-jacking. Spring Security supports to set the needed flag in the HTTP request[15]. "CC.5.2" advocates to enable a user's browser XSS filter to protect against cross site scripting attacks. Spring Security supports setting this header similar to enabling CPS. "CC.5.3" refers to technology no used in the CliniScale project, nonetheless the recommendation stays true. Disabling debugging modes is essential when configuring live versions of the system. "CC.5.4" recommends to only allow trusted origins when using Cross-Origin Resource Sharing (CORS). Setting this option is supported by the Spring Security framework. "CC.5.5" advocates to enable request validation to prevent adversaries from being able to execute specific script-injection attacks. The Spring Data frameworks supports this feature by implementing the Validator interface [16]. "CC.5.6" and "CC.5.10" recommend to only use up-to-date javascript libraries and reference them only from trusted origins and, when possible, over a HTTPS secured connection. "CC.5.7" advises to disable content sniffing for web browsers. This option can be set using the Spring Security framework in a similar way to enabling CSP

[15] Spring Security Headers: https://docs.spring.io/spring-security/site/docs/3.2.0.CI-SNAPSHOT/reference/html/headers.html. (Online; last accessed: November 18, 2019)

[16] Spring Data Rest Validator: https://docs.spring.io/spring-data/rest/docs/current/reference/html/#validation. (Online; last accessed: November 18, 2019)

and XSS filters. ”CC.5.8” recommends encrypting sensitive data, such as usernames and passwords, in configuration files. The Spring Boot framework can be used together with the tool Jasypt (Java Simplified Encryption) to encrypt sensitive data in configuration files[17]. ”CC.5.9” recommends preventing click-jacking attacks. Spring Security enables this feature per default. In case the option has to be set manually, the process is similar to setting the other HTTP request headers.

6.3.6. Cryptography

Control classes of this category give recommendations on which protocols to use to perform cryptography and how they should be configured. There are six control classes:

- CC.6.1: Use only approved symmetric block ciphers and key lengths
- CC.6.2: Use approved block cipher modes and initialization vectors for symmetric ciphers
- CC.6.3: Use approved asymmetric algorithms, key lengths, and padding
- CC.6.4: Use approved random number generators
- CC.6.5: Do not use symmetric stream ciphers
- CC.6.6: Use approved MAC/HMAC/keyed hash algorithms

The implementation of these control classes applies to both the back end and the mobile application. The BSI TR-02102-1 and TR-02102-2 give state of the art recommendations on which protocols, cipher-suites and configurations to use. For all of these control classes, these documents provide valuable guidance. The Spring Security framework offers support for most protocols, cipher-suites and configurations.

To ensure the secure storage of security keys on mobile devices, a Trusted Execution Environment (TEE)[18] should be used. A TEE is a secure area of the main processor guaranteeing data and code loaded into it to be secure regarding confidentiality and integrity. Android 4.3 (API level 18)[19] and higher support the implementation of the Android Keystore[20]. Android Keystore supports most of the cryptographic algorithms listed in the technical guidelines of the BSI.

[17] Spring Boot Configuration with Jasypt: https://www.baeldung.com/spring-boot-jasypt. (Online; last accessed: November 18, 2019)
[18] Trusted Execution Environment: https://www.trustonic.com/news/technology/what-is-a-trusted-execution-environment-tee/. (Online; last accessed: November 18, 2019)
[19] Android 4.3 Jelly Bean: https://www.android.com/intl/de_de/versions/jelly-bean-4-3/. (Online; last accessed: November 18, 2019)
[20] Android Keystore: https://developer.android.com/training/articles/keystore. (Online; last accessed: November 18, 2019)

6.3.7. Exception Management

Exception Management describes handling of software failures, be it forced or accidental, in a secure manner. The five control classes of this category are the following:

- CC.7.1: Ensure that proper exception handling is done in ASP.NET Web API
- CC.7.2: Do not expose security details in error messages
- CC.7.3: Implement Default error handling page
- CC.7.4: Set Deployment Method to Retail in IIS
- CC.7.5: Exceptions should fail safely

The implementation of these controls concerns the back end and the mobile application. Most important of these control classes is ”CC.7.3”. It advises to implement a default error handling page which does not divulges any, possible critical to security, information. Possible failure trigger can be logged server-side but should never be shown to a user to prevent possible adversaries from obtaining information they should not have access to. Spring Boot supports creating and using custom error pages. ”CC.7.1” proposes to implement proper exception handling in Web APIs. Per default, the default error handling page described before is what should be shown to a user in case of any exception. ”CC.7.2” advocates to omit sensitive data, such as server names, procedures, etc., from error messages to prevent an adversary from obtaining any additional information. In case of an exception, security details, such as the stack trace or details of failed procedures, should be logged. ”CC.7.4” recommends setting possible deployment methods to retail when an application is deployed in a live environment. Different settings, for example debug mode, may lead to information disclosure. ”CC.7.5” advocates to implement a proper exception handling throughout the entire code base. The Spring frameworks offer a lot of functionality on the topic of exception handling[21].

6.3.8. Input Validation

This category consists of control classes defining how to filter, scrub or reject the input to the application before further processing. This category consists of 13 control classes:

- CC.8.1: Ensure that each page that could contain user controllable content opts out of automatic MIME sniffing
- CC.8.2: Ensure appropriate controls are in place when accepting files from users
- CC.8.3: Ensure that type-safe parameters are used in Web Application for data access

[21] Spring Exception Handling: https://spring.io/blog/2013/11/01/exception-handling-in-spring-mvc. (Online; last accessed: November 18, 2019)

- CC.8.4: Encode untrusted web output prior to rendering
- CC.8.5: Perform input validation and filtering on all string type Model properties
- CC.8.6: Do not assign DOM elements to sinks that do not have inbuilt encoding
- CC.8.7: Validate all redirects within the application are closed or done safely
- CC.8.8: Implement input validation on all string type parameters accepted by Controller methods
- CC.8.9: Avoid using Html.Raw in Razor views
- CC.8.10: Ensure that model validation is done on Web API methods
- CC.8.11: Implement input validation on all string type parameters accepted by Web API methods
- CC.8.12: Ensure that type-safe parameters are used in Web API for data access
- CC.8.13: Sanitization should be applied on form fields that accept all characters e.g, rich text editor

The implementation of control classes of this category concerns the back end. The category can be further split into two sub-categories:

User Input Validation Control classes of this category focus on validating a given users input. ”CC.8.5”, ”CC.8.8”, ”CC.8.10” and ”CC.8.11” all recommend input validation on any string type parameter in the Web API, controller classes or model properties. A whitelist validation strategy is recommended, meaning that per default all inputs are invalid and only expected values are accepted by using regular expressions. Spring frameworks support various ways to perform input validation further described in the documentation[22]. ”CC.8.2” advocates to implement measures to user file upload. Many attacks process is to somehow get code into the system attacked and then execute it. A user file upload can help the attacker perform the first step. Many possible measures exist to prevent this problem: File extension checks, maximum file size limit, validating naming conventions, scan the file with an anti-virus or save the file on a non-system drive. ”CC.8.3” and ”CC.8.12” recommend using type safe parameters to prevent injection attacks. Parameters can be used to enforce length and type constraints on input data. Type safe parameters also prevent code execution, as the database treats them as literal values and not as executable code.

[22] Spring Validation, Data Binding, and Type Conversion: https://docs.spring.io/spring/docs/4.1.x/spring-framework-reference/html/validation.html. (Online; last accessed: November 18, 2019)

Input Validation Configuration Control classes of this category prevent abuse of user input by disabling options per configuration. ”CC.8.1” recommends, similar to ”CC.5.1”, to disable automated MIME sniffing by setting the HTTP header ”X-Content-Type-Options:nosniff”. Spring Security supports setting this header, as described in ”CC.5.1”. ”CC.8.4” advocates to encode untrusted web output to prevent cross-site scripting attacks. To implement this control class, it is recommended to perform input validation and encoding of user input before it is rendering. ”CC.8.6” recommends to not assign untrusted input to DOM elements, since many javascript functions do not perform encoding by default. ”CC.8.7” advocates to whitelist possible redirections to user-supplied locations. This prevents adversaries from possible stealing a user’s logon token to services like Facebook/OAuth/OpenID. ”CC.8.9” advocates to not use any functions that may bypass automatic encoding protection, like the Html-Helper.Raw method, that return not HTML encoded code. ”CC.8.13” recommends performing sanitization on all form fields that accept all characters. Sanitization cleans HTML fragments and prevents cross-site scripting attacks. If proper input validation and output encoding measures are implemented, there is no need for sanitization.

6.3.9. Sensitive Data

The category ”Sensitive Data” consists of control classes securing name giving data by protecting it in memory, over the network or on storage in browser or mobile application. There are seven control classes in this category:

- CC.9.1: Ensure that sensitive content is not cached on the browser
- CC.9.2: Encrypt sections of Web App’s configuration files that contain sensitive data
- CC.9.3: Explicitly disable the autocomplete HTML attribute in sensitive forms and inputs
- CC.9.4: Ensure that sensitive data displayed on the user screen is masked
- CC.9.5: Ensure that sensitive data relevant to Web API is not stored in browser’s storage
- CC.9.6: Encrypt sensitive or PII data written to phones local storage
- CC.9.7: Obfuscate generated binaries before distributing to end users

Implementation of these control classes concerns both the back end and the mobile application. ”CC.9.1” and ”CC.9.5” recommend to not cache any sensitive data in the browser. Cached data, such as username, health data or personal details, can be accessible even after the user logs off. Especially on shared computers these can lead to grave information disclosure. To prevent a browser from caching data, a response

header called ”Cache-Control” can be set to ”no-cache”. The Spring Security framework supports setting this option by using the CacheControl class[23]. ”CC.9.2” recommends encrypting sensitive data in configuration files, such as server addresses and passwords, to prevent adversaries of gaining access to the data. See ”CC.5.8” for an explanation on the implementation. ”CC.9.3” advocates to disable autocomplete on sensitive forms and inputs, such as username/password. By default, autocomplete is on. To disable autocomplete, the attribute ”autocomplete” should be se to ”off” either on the entire form or single input fields. ”CC.9.4” recommends masking sensitive data displayed for a user, preventing adversaries from accessing the data. Masking data has to be taken care off when the data is accepted as input, e.g. by setting the HTML attribute ”input type” to ”password”, and when displayed on the screen, e.g. by displaying only the last digits of a sensitive number such as credit card number. ”CC.9.6” recommends encrypting sensitive information that is saved on the mobile device’s file system. This prevents attackers from accessing the data, e.g. when they root the phone and can read out the local file storage. ”CC.9.7” recommends obfuscating generated binaries, e.g. apk files for Android mobile devices, to prevent adversaries from reverse engineering the assemblies. Tools, such as ProGuard[24], can be used for this purpose.

6.3.10. Session Management

Control classes of this category secure user sessions between the user and the web application. The category consists of five control classes:

- CC.10.1: Applications available over HTTPS must use secure cookies
- CC.10.2: All http based application should specify http only for cookie definition
- CC.10.3: Mitigate against Cross-Site Request Forgery (CSRF) attacks on ASP.NET web pages
- CC.10.4: Set up session for inactivity lifetime
- CC.10.5: Implement proper logout from the application

Implementation of these control classes happens in the back end. ”CC.10.1” recommends using secure cookies on connections using the HTTPS protocol. Secure cookies are cookies with the ”secure” attribute set and are only available when using HTTPS. The Spring Security framework supports setting the ”secure” attribute[25]. ”CC.10.2” recommends making cookies from HTTP based applications unable to be accessible by script by setting the ”httpOnly” attribute. Implementation of this attribute

[23] Spring Security CacheControl: https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/http/CacheControl.html. (Online; last accessed: November 18, 2019)
[24] ProGuard: https://www.guardsquare.com/en/products/proguard. (Online; last accessed: November 18, 2019)
[25] Control the Session with Spring Security: https://www.baeldung.com/spring-security-session. Last access: (Online; last accessed: November 18, 2019)

is similar to the ”secure” attribute presented in ”CC.10.1”. ”CC.10.3” advocates to implement measures preventing CSRF attacks. Spring Security has a build in measure against this kind of attacks implementing the Token Synchronizer Pattern[26]. ”CC.10.4” recommends implementing a session timeout as a measure to prevent adversaries from gaining access to an users account who has not logged off. Spring Security supports implementation of a custom session timeout timer[27]. ”CC.10.5” recommends destroying a user’s session on user log out and reset and nullify the session cookie and authentication cookie values. The Spring Security framework offers the needed functionality[28].

[26] Spring Security CSRF: https://docs.spring.io/spring-security/site/docs/3.2.0.CI-SNAPSHOT/reference/html/csrf.html. (Online; last accessed: November 18, 2019)

[27] Control the Session with Spring Security: https://www.baeldung.com/spring-security-session. (Online; last accessed: November 18, 2019)

[28] Spring Security Logout: https://www.baeldung.com/spring-security-logout. (Online; last accessed: November 18, 2019)

7. Results

The chapter *Results* consists of the sections *Summary* and *Limitations*. The first section, *Summary*, presents the findings of this thesis. In the second section, *Limitations*, aspects that are not part of this thesis are presented.

7.1. Summary

In the context of this thesis a security risk assessment is performed and in the process a security concept for the secure transfer of sensitive personal data is designed and evaluated.

The MoRA methodology is modified to use the Microsoft STRIDE as threat catalogue and the controls defined in the Microsoft Threat Modeling Tool as control catalogue. Also, three activities of the MoRA methodology are altered in order to support the import of threats and controls generated by the Microsoft Threat Modeling Tool: *Determine Protection Needs*, *Analyze Threats* and *Establish Controls*. As the MoRA methodology is built in a modular way, consisting of various activities and work products, alteration of single activities is possible if the consumed and produced artifact are the same.

The statutory regulations implemented by the GDPR are analyzed. The articles and recitals with legal consequences regarding this thesis are identified and statutory requirements to the security concept designed in this thesis are extracted and defined as requirements. Further, it is analyzed which requirements are covered in the process of performing a security risk assessment.
Next, the process of the CliniScale system is analyzed and modeled. The three parties involved in the process, the *CliniScale Environment*, the *Trial Executor* and the user of the *CliniScale Application* for mobile devices, are identified. The process is further analyzed to create an abstract workflow used to model the system architecture for the security risk assessment at a later step.
Last, a reliable source for up-to-date cryptographic recommendations is identified in the BSI technical guidelines TR-02102-1 and TR-02102-2.

To perform a security risk assessment, preliminary tasks must be performed.
First, the *Assessment Model* is configured to fit the needs of a security risk assessment investigating the CliniScale project. In the process, six security goal classes, three levels of damage potentials, two damage criteria partitioned in three damage subclasses, four levels of required attack potentials and three risk levels are defined.

The next step in preparation for the risk assessment is to model the system architecture as *System Under Development*. Following the definition of the CliniScale pro-

cess analyzed in *CliniScale System Process*, functions, components, data elements and channels are defined according to the *Document System Under Development* activity and the *System Under Development (SUD)* is created.
To integrate the Microsoft tools into the MoRA methodology, the *SUD* is modeled in the Microsoft Threat Modeling Tool.
Threat and control catalogues are synthesized of the treats and controls generated in the Microsoft Threat Modeling Tool. A set of the generated threats is categorized using the STRIDE threat model and imported into the risk assessment as threat catalogue. The threat catalogue created consists of 44 threat classes. Similar to threats, controls are categorized using the ten categories defined by the Microsoft Threat Modeling Tool and imported into the risk assessment as control catalogue. The control catalogue consists of 82 control classes.
With the creation of the artifacts *Threat Catalogue* and *Control Catalogue* and the work product *System Under Development*, activities leading to the work product *Risk Assessment* can be performed.

In order to perform the security risk assessment and create the work product *Risk Assessment*, the four activities *Determine Protection Needs*, *Analyze Threats*, *Analyze Risks* and *Establish Controls* are performed.
In the process of performing *Determine Protection Needs* 54 security goals are defined, 24 security goals concerning data elements and 30 security goals concerning data flow elements. Of these security goals, 27 have the damage potential ”High”, 15 the damage potential ”Moderate” and twelve the damage potential ”Low”.
Analyze Threats is the next activity defined in the MoRA methodology. In the process, 38 threats split in the six STRIDE categories are defined.
The next activity is *Analyze Risks* to define of risk elements. In the process, 38 risk elements are defined. Of these, 36 cause a risk level of ”High” and two cause a risk level of ”Moderate”. Since the average risk level is too high to be acceptable, controls mitigating the threats must be identified.
To mitigate the high risk levels, the activity *Establish Controls* is performed. Split on ten categories, 72 controls are defined. The mitigating effect of these controls is notable in the risk elements: 15 risk elements now have a risk level of ”Moderate” and 23 risks a risk level of ”Low”. Discussing the resulting risk levels and the implementation of the controls, it is determined that there is no need for a second iteration of the risk assessment as the risk levels are acceptable.

At last, an implementation guideline of the identified controls for the CliniScale system is created. The current implementation of the CliniScale system is analyzed by identifying the system architecture and the technologies used.
Categorized by the ten categories defined in the Threat Modeling Tool, for all 72 control classes practical implementation suggestions, if possible, using the already used frameworks and libraries of the current implementation of the CliniScale system, are given.

7.2. Limitations

The scope of this thesis being the design of a security concept for the transfer of sensitive data, it leaves open the question of how to store them locally in a secure manner. Secure storage, depending on the underlying system, is as important as securing the data transfer. Especially the storage of sensitive personal data and secret keys used for encryption possess a critical value.

While articles of the General Data Protection Regulation regarding the transfer of sensitive data have been identified, many more articles would apply to an information system in the health domain. It is subject of further research to identify these articles and their requirements. Issues, such as the requirement of a data protection impact assessment for article 35[9] or the implementation of a user's right to access, erasure or restriction defined in the articles 15[3], 17[4] and 18[5], have to be identified.

The Microsoft STRIDE threat model and the mitigation categories of the Microsoft Threat Modeling Tool offer a good overall coverage, being catalogues that are universally applicable. Although this implies that they may lack domain specific knowledge. Threats that are specific to the domain of information systems in healthcare may not be covered by the used catalogues and therefore the resulting risk level could be false. Another limitation of using catalogues based on the STRIDE model is the missing property of a threat's probability. Instead of estimating the required attack potential, the quality of the catalogues could have been improved by applying a risk rating model.

Configuring the *Assessment Model* as a preliminary step for the security risk assessment is a task based on the operator's experience. Some details, like defining security goal classes, are predefined based on the usage of the Microsoft STRIDE threat model. Definition of the damage classes and its subclasses lies in the experience of the operator. The MoRA methodology doesn't give a clear artifact on which classes to use, as it can vary from one domain to another. It is assumed that, besides the risk of a loss of self-determination and financial or reputational damage, there is no further damage class which warrants to be considered.

8. Conclusion

The chapter *Conclusion* covers the sections *Discussion of Results* and *Future Work*. In *Discussion of Results* the results of the previous chapter *Results* are discussed. In the second section, *Future Work*, gives impulse for further research.

8.1. Discussion of Results

The results of the security risk analysis show that the security concept is applicable for use in a client-server infrastructure with a mobile application and complies with the requirements of the General Data Protection Regulation (GDPR). The use of the security concept is therefore not limited to the use in the CliniScale project, but can be applied to any system with a similar infrastructure. Due to the determination of appropriate security measures depending on the potential risk of the examined data, relevant regulations of the GDPR have been fulfilled.

The modular approach of the MoRA methodology allows for a successful integration of the Microsoft Threat Modeling Tool in the form of threat and control catalogues. Threats generated in the Microsoft Threat Modeling Tool and their suggested controls are easily integrated into the MoRA methodology. The lack of a defined required attack potential is adjusted by estimating the values based on the experience of the security analyst.

The regulations implemented by the GDPR regarding this thesis have been identified. From a security point of view, they are vague on the topics of determining the possible risks associated with personal data and the actual implementation of measurements. The unspecific regulations regarding encryption, specifically which standards or protocols to implement, creates the impression that the GDPR tries to dodge the responsibility by putting it in the hand of the controller. External help, in the form of the technical guidelines of the German Federal Office for Information Security, has to be counseled to reach the requested level of security. A regulation regarding the security and privacy of natural persons should have made a greater effort in defining technical guidelines to follow when implementing required security measurements.

8.2. Future Work

Open topics when it comes to compliance with the GDPR have to be discussed. The process of registration of a user and obtaining his consent for the gathering and pro-

cessing of personal data has to be defined. User roles and access rights should be determined to enable the implementation of role based access. Mechanism to detect intrusion in order to notify authorities and users of possible disclosure of personal data, as required by the GDPR, have to be implemented.

The implementation guideline of identified controls has to be applied to the current implementations of the CliniScale system, both server side and to the mobile application.

In order to guarantee a sufficient level of security to the entire system, a security concept regarding the storage of critical information, such as sensitive personal data or secret keys, has to be developed for both the application on a server infrastructure and on mobile applications.

New threat and control catalogues, with the focus on application to systems similar to CliniScale, can be developed. Threats specific to the domain of mobile healthcare applications have to be identified and categorized. Further, matching controls that mitigate the effect of identified threats should be identified in order to create a control catalogue. If said catalogues are developed, a new security risk assessment using them should be performed.

Adopting secure software engineering processes, such as Secure Software Development Lifecycles (SSDLC) and Security Maturity Models (SMM), help improve the security level of designed, implemented and evaluated software. SSDLC are models that define the process of software development, with a focus on security. Their goal is to improve the level of security by ensuring that security assurance activities, such as risk assessments, penetration testing or code review, are an integral part of the development effort. SMM assess the level of implementation of security processes and gives a guideline on how to further improve these.

9. References

[1] **BSI TR-02102-1**. *(Online; Last accessed November 18, 2019)*. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf.

[2] **BSI TR-02102-2**. *(Online; Last accessed November 18, 2019)*. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-2.pdf.

[3] **General Data Protection Regulation Article 15**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-15-gdpr/.

[4] **General Data Protection Regulation Article 17**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-17-gdpr/.

[5] **General Data Protection Regulation Article 18**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-18-gdpr/.

[6] **General Data Protection Regulation Article 25**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-25-gdpr/.

[7] **General Data Protection Regulation Article 32**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-32-gdpr/.

[8] **General Data Protection Regulation Recital 32**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/recitals/no-32/.

[9] **General Data Protection Regulation Article 35**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-35-gdpr/.

[10] **General Data Protection Regulation Article 37**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-37-gdpr/.

[11] **General Data Protection Regulation Article 4**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-4-gdpr/.

[12] **General Data Protection Regulation Article 40**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-40-gdpr/.

[13] **General Data Protection Regulation Article 42**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-42-gdpr/.

[14] **General Data Protection Regulation Recital 51**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/recitals/no-51/.

[15] **General Data Protection Regulation Article 6**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-6-gdpr/.

[16] **General Data Protection Regulation Article 7**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-7-gdpr/.

[17] **General Data Protection Regulation Article 84**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-84-gdpr/.

[18] **General Data Protection Regulation Article 9**. *(Online; Last accessed November 18, 2019)*. URL: https://gdpr-info.eu/art-9-gdpr/.

[19] **D. Angermeier, A. Nieding, and J. Eichler**. *Supporting Risk Assessment with the Systematic Identification, Merging, and Validation of Security Goals*. Ed. by J. Großmann, M. Felderer, and F. Seehusen. Cham: Springer International Publishing, 2017, pp. 82–95. ISBN: 978-3-319-57858-3.

[20] **D. Angermeier, A. Nieding, and J. Eichler**. "Systematic Identification of Security Goals and Threats in Risk Assessment". In: *Softwaretechnik-Trends* 36.3 (2016).

[21] **R. Canettin and H. Krawczyk**. "Analysis of key-exchange protocols and their use for building secure channels". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2001, pp. 453–474.

[22] **BSI IT-Grundschutz Catalogues**. *(Online; Last accessed November 18, 2019)*. URL: https://www.bsi.bund.de/EN/Topics/ITGrundschutz/ITGrundschutzCatalogues/itgrundschutzcatalogues_node.html.

[23] **E. Chin, A. P. Felt, V. Sekar, and D. Wagner**. "Measuring user confidence in smartphone security and privacy". In: *Proceedings of the eighth symposium on usable privacy and security*. ACM. 2012, p. 1.

[24] **K. Cios, B. Krawczyk, J. Cios, and K. Staley**. "Uniqueness of Medical Data Mining: How the new technologies and data they generate are transforming medicine". In: *arXiv preprint arXiv:1905.09203* (2019).

[25] **T. Cooijmans, J. de Ruiter, and E. Poll**. "Analysis of secure key storage solutions on android". In: *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*. ACM. 2014, pp. 11–20.

[26] **J. Eichler and D. Angermeier**. "Modular Risk Assessment for the Development of Secure Automotive Systems". In: *31. VDI/VW-Gemeinschaftstagung Automotive Security* (2015).

[27] **S. Fahl, M. Harbach, T. Muders, L. Baumgärtner, B. Freisleben, and M. Smith**. "Why Eve and Mallory love Android: An analysis of Android SSL (in) security". In: *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM. 2012, pp. 50–61.

[28] **M. de Gramatica, K. Labunets, F. Massacci, F. Paci, and A. Tedeschi**. "The role of catalogues of threats and security controls in security risk assessment: an empirical study with ATM professionals". In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer. 2015, pp. 98–114.

[29] **H. Krawczyk, K. Paterson, and H. Wee**. "On the security of the TLS protocol: A systematic analysis". In: *Annual Cryptology Conference*. Springer. 2013, pp. 429–448.

[30] **BSI TR-02102 Cryptographic Mechanisms**. *(Online; Last accessed November 18, 2019)*. URL: https://www.bsi.bund.de/EN/Publications/TechnicalGuidelines/tr02102/tr02102_node.html.

[31] **M. Meingast, T. Roosta, and S. Sastry**. “Security and privacy issues with health care information technology”. In: *Engineering in Medicine and Biology Society, 2006. EMBS’06. 28th Annual International Conference of the IEEE*. IEEE. 2006, pp. 5453–5458.

[32] **Microsoft Threat Modling Tool Mitigations**. *(Online; Last accessed November 18, 2019)*. URL: https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-mitigations.

[33] **General Data Protection Regulation**. *(Online; Last accessed November 18, 2019)*. URL: https://eugdpr.org/.

[34] **Microsoft STRIDE**. *(Online; Last accessed November 18, 2019)*. URL: https://docs.microsoft.com/en-gb/azure/security/develop/threat-modeling-tool-threat.

[35] **Microsoft Threat Modeling Tool**. *(Online; Last accessed November 18, 2019)*. URL: https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool.

[36] **Vivy Security Whitepaper**. *(Online; Last accessed November 18, 2019)*. URL: https://www.vivy.com/fileadmin/user_upload/20181018_Vivy_Frauenhofer_Institut_Whitepaper_Security.18.pdf.

[37] **W. Wilkowska and M. Ziefle**. “Privacy and data security in E-health: Requirements from the user’s perspective”. In: *Health informatics journal* 18.3 (2012), pp. 191–201.

[38] **I. Williams and X. Yuan**. “Evaluating the effectiveness of Microsoft threat modeling tool”. In: *Proceedings of the 2015 information security curriculum development conference*. ACM. 2015, p. 9.

A. Source Code of Implementation

DVD A DVD is added to this Master thesis. It contains a PDF version of this thesis, the source code of the security risk assessment and the Microsoft Threat Modeling Tool model, the created threat and control catalogues as Excel files and an export of the results of the security risk assessment as HTML file.

Repositories The source code of this thesis is accessible on the GitLab site of the Freie Universität Berlin under the link below:

https://git.imp.fu-berlin.de/fub-agdb/cliniscale-secure-transfer-of-medical-data