UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Cybersecurity Curriculum

Maria Pibilota Murumaa

Designing a security sensitive self-assessment framework

Master's Thesis (21 ECTS)

Supervisors: Mari Seeba, MSc

Tarmo Oja, MSc

Tartu 2023

Designing a security sensitive self-assessment framework

Abstract:
The Estonian Information Security Standard (E-ITS) development has brought the need to evaluate organisation's information security state. However, collecting data about security measures must be handled securely. This thesis aims to design a security sensitive Self-Assessment Framework (SAF) for collecting answers to F4SLE (Framework for Security Level Evaluation).

To propose the SAF design, a similar tool comparison, requirement analysis, and three design iterations were performed. The final design included a web-based user interface for collecting aggregated results and server-based administrative functionality for benchmark calculations and visualisation. In addition, a limited version of the SAF was implemented to conduct a pilot in Estonia and the Czech Republic.

The SAF validation consists of two parts. Firstly, threat analysis is conducted to evaluate the framework's security posture and identify additional requirements. Secondly, the pilot participants are asked to assess the framework to validate design decisions.

The proposed security sensitive SAF design can be generalised to other 4-level self-assessment tools. The framework is suitable for conducting threat audits or validating newly developed risk assessment frameworks.

Keywords:
F4SLE, self-assessment framework design, threat analysis, security sensitive self-assessment framework

CERCS: T120 - Systems engineering, computer technology

Turvatundliku enesehindamise raamistiku projekteerimine

Lühikokkuvõte:

Eesti Infoturbestandardi (E-ITS) loomisega kaasnes vajadus hinnata organisatsioonide infoturbe seisundit. Turvameetmeid kirjeldavate andmete kogumist tuleb aga käsitleda turvaliselt. Selle töö eesmärk on F4SLE (*Framework for Security Level Evaluation*) vastuste kogumiseks projekteerida turvatundlik enesehindamise raamistik (SAF).

SAF loomiseks viidi läbi järgnevad sammud: sarnaste tööriistade võrdlus, nõuete analüüs ja kolm projekteerimise iteratsiooni. Raamistiku lõplik tulem koosneb veebipõhisest kasutajaliidesest, millega kogutakse asutuste tulemused, ja serveripõhisest haldusfunktsioonist mvõrdlusaluse arvutamiseks ja visualiseerimiseks. Lisaks arendati piirangutega SAF versioon, et läbi viia piloot mõõtmised Eestis ja Tšehhis (Lõuna-Moravias).

Raamistiku valideerimine koosnes kahest osast. Esmalt viidi läbi ohuanalüüs, et hinnata raamistiku turvatundlikkust ja täiendada raamistikule seatud nõudeid. Teiseks paluti piloodis osalejatel hinnata raamistikku, et valideerida projekteerimisotsused.

Loodud turvatundliku enesehindamise raamistikku saab üldistada 4-tasandiliste küsimustike kasutamiseks. Arendatud raamistikku saab kasutada ohuauditite läbiviimiseks või väljatöötatud riskihindamisraamistike valideerimiseks.

Võtmesõnad:

F4SLE, enesehindamise raamistiku projektsioon, ohu analüüs, turvatundlik enesehindamise raamistik

CERCS: T120 - Süsteemitehnoloogia, arvutitehnoloogia

Acknowledgments

I am extremely grateful to my supervisors, Mrs. Mari Seeba and Mr. Tarmo Oja. Thank you for being insightful, dedicated, and patient. I appreciate all the experiences, knowledge, and wisdom.

This endeavor would not have been possible without my family. Thank you for always believing in me and supporting me throughout my academic journey.

I would also like to thank the Cybernetica AS team! The discussions and advice helped to shape this thesis.

This work is part of the Cyber-security Excellence Hub in Estonia and South Moravia (CHESS) project funded by the European Union under Grant Agreement No. 101087529. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Contents

1 Introduction

As of December 2022 the Estonian Cybersecurity Act has been commenced [1]. The act's requirements are defined by the Estonian Information Security Standard (E-ITS). The standard aims to develop the information security management in both small and large organisations in Estonia [2]. Around 3500 organisations are required to implement and audit the standard within three years [3].

The E-ITS development brought the need to evaluate organisation's overall information security state. No comprehensive information security measuring and result comparison has been conducted in Estonia [4]. Thus, Mari Seeba et al. [5] have created an E-ITS-based Framework for Security Level Evaluation (F4SLE) that allows organisations to self-assess their current implementation state. Collecting organisations' F4SLE results would create a benchmark that can be used for national or domain based benchmarking.

Using the created benchmark it is possible to assess the national security posture and take necessary steps to correct found shortcomings, for example, by arranging mandatory training or exercises. The benchmark data can be used to see the overall implementation state of E-ITS in Estonia and help to shape legislation.

However, collecting data about security measures must be treated confidentially as information about the organisation's current security state, used technologies or management styles can lead to exploits of various severity and kind. The Estonian Public Information Act [6] §35 point 9 states that "information including a description of security systems, security organisations or security measures should be considered as restricted information." Meaning any information related to organisations security should be handled as classified information that is intended for internal use only.

This thesis will focus on proposing a security sensitive self-assessment framework design for collecting orgnaisations' F4SLE results without revealing the organisations exact security measures.

1.1 Motivation

The thesis is motivated by the need to validate Mari Seeba et al. [5] developed Framework for Security Level Evaluation (F4SLE). The authors state that in order to validate the developed framework and derived a benchmark, more responses to F4SLE need to be gathered.

By designing a low-entry barrier self-assessment framework, it is possible to widen the

number of respondents of F4SLE and collect data to validate Seeba's created framework. Furthermore, the wide adaptation of a security maturity self-assessment tool could improve the overall security posture, help organisations to map their current condition, and encourage the overall discussion about information security state.

1.2 Research problem

This thesis aims to design a security sensitive self-assessment framework. Meaning, the designed framework should preserve the collected organisation's security data secrecy. To achieve this goal the following primary research question (PRQ) is proposed: **How should the security sensitive self-assessment framework design look like?**

To address the primary research question the following questions should be answered:

SRQ1: What are the requirements to implement the proposed framework design?

SRQ2: What is the security posture of the proposed framework design?

1.3 Research Method

To conduct the thesis, Design Science and the Generate/Test Cycle [7] approach was utilized. The overview of the thesis is shown in Fig. 1. Firstly, the problem was identified and related objectives derived. To create the design of the framework three design iterations were performed. In each iteration decisions regarding the design were evaluated and implemented. The created design was validated by conducting a threat analysis and a pilot in Estonia and Czech Republic. Lastly, all findings were evaluated and communicated.

1.4 Contribution

The following thesis is part of the Cyber-security Excellence Hub in Estonia and South Moravia (CHESS) project. The outcome of this thesis included a design of a security sensitive self-assessment framework for gathering responses to F4SLE. To validate the proposed design a threat analysis was conducted.

In addition, a limited proof of concept (PoC) framework was implemented. The PoC implementation was utilized to launch a pilot in Estonia and Czech Republic to collect organisation's security maturity levels. The pilot results are published in ARES 2023 SP2I workshop paper "Security level evaluation with F4SLE" [4].

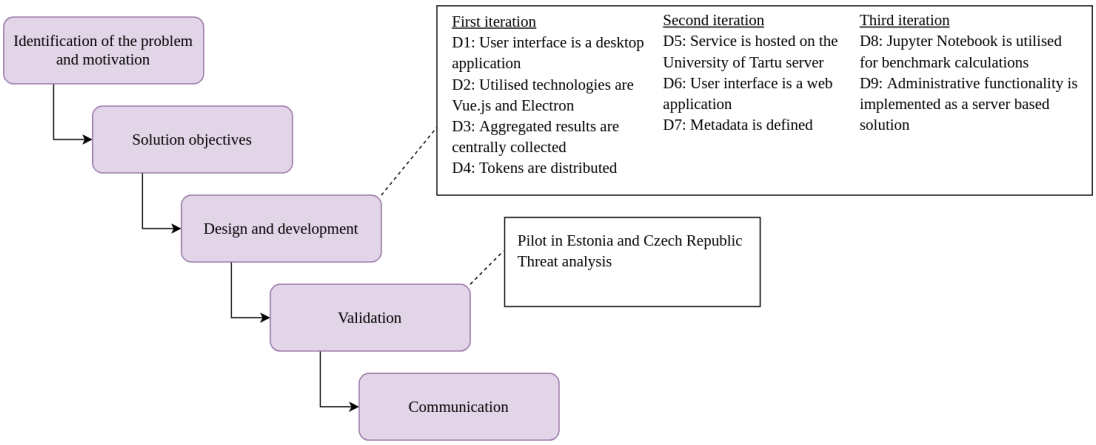

Figure 1. Thesis research method

2 Background

The following chapter gives an overview of the Estonian Information Security Standard (E-ITS) [8] and the Framework for Security Level Evaluation (F4SLE) [5] to describe the principles affecting the Self-Assessment Framework's (SAF) design and it's requirements.

A threat analysis is conducted to validate the created design. Thus, the chapter introduces the primary methodologies used to conduct the threat analysis described in Chapter 5.6. Lastly, the background chapter conducts an analysis of similar tools and compares their weaknesses and strongpoints. The results of the self-assesment tools comparison can be used as requirements for the developed SAF (**SRQ2**).

2.1 Estonian Information Security Standard (E-ITS)

The Estonian Information Security Standard (E-ITS) aims to provide organisations in all areas of activity with an Estonian tool to handle information security [9]. The standard divides protected objects and processes into ten elementary threat catalog modules [9]. The modules describe the most common threats and corresponding industry-specific security measures based on best practises. The modules are divided into two groups, process modules and system modules [10]. Process modules focus on the involvement of management and consist of Information Security Management system (ISMS), Organisation and Personel (ORP), Concepts (CON), Operation (OPS), Detection and Reaction (DER). System modules are more technical and consist of Applications (APP), IT Systems (SYS), Industry IT (IND), Network and Communication (NET), Infrastructure (INF).

2.2 Framework for Security Level Evaluation (F4SLE)

Mari Seeba et al. [5] have created a Framework for Security Level Evaluation (F4SLE) that is based on the Estonian Information Security Standard (E-ITS) [9] and compliant with ISO/IEC 27002 [11] controls. The created framework is a questionnaire that requires the user to evaluate described attributes implementation state in their organisation. Based on the user provided answers benchmarks are calculated. F4SLE provides consistent and comparable results as it can be updated yearly as E-ITS changes. Currently, the 2022 E-ITS version is valid.

The first F4SLE version was represented as a Microsoft Word document [12] where participants had to mark each statement with the corresponding answer color: red, orange, yellow, or green. The second and current F4SLE representation is a Microsoft Excel

file [13] where the user should fill the answer column with numerical values from 1 to 4. In both cases, all security-related data is sent outside the organisation: the detailed answer file should be encrypted by the user and emailed to the framework manager to create a benchmark.

The current approach for F4SLE result gathering has flaws. To receive answers, we expect the participant to perform additional actions. This approach is unreliable as performing additional steps may lessen the user's interest to participate. Furthermore, participants could forget to encrypt the file or send it to the wrong recipient.

Gathering files that contain detailed assessments of organisations' security measures poses a security issue. Firstly, the Estonian Public Information Act §35 point 9 [6] states that information about organisation security measures should be used only internally. Therefore, detailed assessments of security measures should not leave the user's computer. Secondly, if a malicious advisory gains access to the detailed answer file, they can utilize its content to craft highly targeted and effective attacks that may result in various damages (e.g., data, monetary, reputation).

This paper addresses the issues and proposes a framework to resolve the current shortcomings.

2.2.1 Framework manager role

The framework manager is the term used to represent the person responsible for creating and updating the framework and its content. For example, the F4SLE framework manager formed the questionnaire by deriving around 200 attributes from 1500 E-ITS security controls [14].

Security methods and regulations constantly change to keep up with new technologies and shifts in the threat landscape. For example, E-ITS controls change each year: 100 new controls are added, and around 100 controls are changed or substituted [14]. The framework manager is responsible for reflecting the changes in the framework. Before launching the updated framework, the framework manager is responsible for validating the changes by organising pre-tests [14].

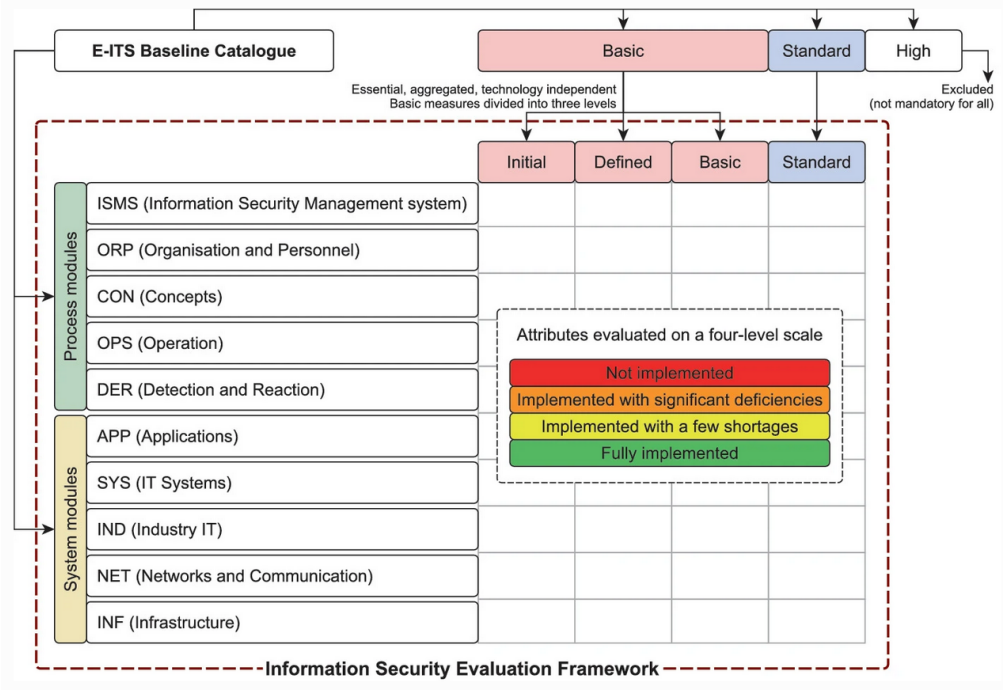

Figure 2. Process of maturity model framework design and development [5].

2.2.2 Questionnaire

The F4SLE questionnaire consists of around 200 statements [13] which are divided into 10 module dimension groups derived from E-ITS Baseline Catalogue. Fig. 2 illustrates the Baseline Catalogue conversion process to F4SLE dimension groups.

Each question i.e. attribute is assigned a corresponding security maturity level – Initial, Defined, Basic, Standard. Each level contains at least one attribute. A module contains statements from all four maturity levels. The module can belong to either to the process modules group (ISMS, ORP, CON, OPS, DER) or to the system modules group (APP, SYS, IND, NET, INF) [4] as described in Chapter 2.1.

Each statement can be answered with the following options [13]:

- **No answer** – the statement does not apply to the organisation (e.g. no vehicles, robots, process management systems connected to bar code readers are used);
- **1** – the content of the attribute could be the goal, but has not yet reached it;
- **2** – attribute is partially in accordance with the description of the situation, but still with significant shortcomings;

- **3** – the attribute is consistent with your organization, but with some shortcomings;
- **4** – the attribute is completely true in the context of your organization.

In addition to the detailed answers, additional metadata is collected from the participant: name of the respondent, respondent role in organisation, date of completion, organisation, the organisastion's domain [school, university, hospital, family medical center, state office, etc.], the number of computer workplaces belonging to the protected area of the organisastion, the number of geographical locations of the organisastion that depend on ITK [e.g., branch offices], time taken to respond to the questionnaire.

2.2.3 F4SLE Benchmark

A benchmark is a set of values that can be used for measuring and comparison [15]. The F4SLE benchmark allows organisations to compare their information security results to other organisations' average values, track progress and management life cycle, and prioritise the implementation of security measures [4]. Furthermore, creating a national benchmark can help policymakers to observe the overall E-ITS implementation state and provide input for funding or training allocation.

To create such a benchmark, participants' evaluations of the questionnaire attributes are gathered. Each attribute can be evaluated with a value from 1 to 4. Based on the F4SLE input, 4 x 10 benchmarks are calculated: Initiated, Defined, Basic and Standard benchmark for each module. The benchmarks are calculated by finding each level's average value. An overall benchmark for every module is calculated using the following formula: $(SUM) - 4)/4$, where SUM is the total of the level benchmark value. The domain benchmark value ranges from 0 to 3 and can be interpret as shown in Table 1.

Table 1. F4SLE benchmark values [13].

Range	Label	Definition
<0.75	Initiated	The need to deal with information security has been acknowledged and addressed.
>=0.75 and <1.5	Defined	Formal processes have been agreed, and the necessary information security supporting documents have been prepared.

>=1.5 and <2.25	Basic	Practical basic activities have been implemented to manage information security.
>2.25	Standard	There are clear organisational policies and principles. Activities are standardised, documented, regular and monitored. There is ongoing monitoring and improvement.

2.3 Threat analysis

A threat analysis aims to identify and analyse potential security or privacy threats to mitigate potential harm to the system [16]. Threat analysis can be achieved through threat modeling: it is possible to anticipate possible threats to systems not built yet by abstracting details [17]. Meaning it is possible to build a focused defense for the system and its assets by analysing possible generalized types of threats.

There are numerous approaches to threat analysis [16, 17]. To conduct a threat analysis on the proposed design, a combination of Oja's [18] Method for Identifying Potential Weaknesses and Muckin et al. [19] IDDIL/ATC discovery phase is used for a systematic approach. Both methods require decomposing the system to find possible threats and attack vectors. All identified possible threats will be categorized using the STRIDE model [20].

The Method for Identifying Potential Weaknesses [18], IDDIL/ATC discovery phase [19], and STRIDE model [20] were chosen as they allow a systematic approach for threat analysis. The listed methods provide an "agile" approach [18]: moving across the method phases and supplementing the results is possible. The combination is fitting for analysing designs as it can be utilised to model threats and abstract details.

STRIDE is an acronym developed by Microsoft that helps to model threats [20]. The model can be applied to group threats that affect system components, functions, or data into categories. Furthermore, the model can be used to propose possible security controls or mitigations for identified threats. In Table 2, the STRIDE threat categories and properties are explained.

Table 2. STRIDE threat categories and security properties [19].

STRIDE	Threat category	Property
S	Spoofing	Authentication
T	Tampering	Integrity
R	Repudiation	Non-repudiation
I	Information Disclosure	Confidentiality
D	Denial of Service	Availability
E	Elevation of Privilege	Authorization

Oja's [18] Method for Identifying Potential Weaknesses proposes a layered approach to threat analysis. Illustrated in Fig. 3, the method provides six categories to decompose the system. Creating a trust relationship model, asset threat profile, protocol flow, and component data flow model is a systematic approach to map positive scenarios in the system. When conducting a threat analysis, attack scenarios can be generated by finding possible threats and risks for the positive scenarios and by moving between the created layers.

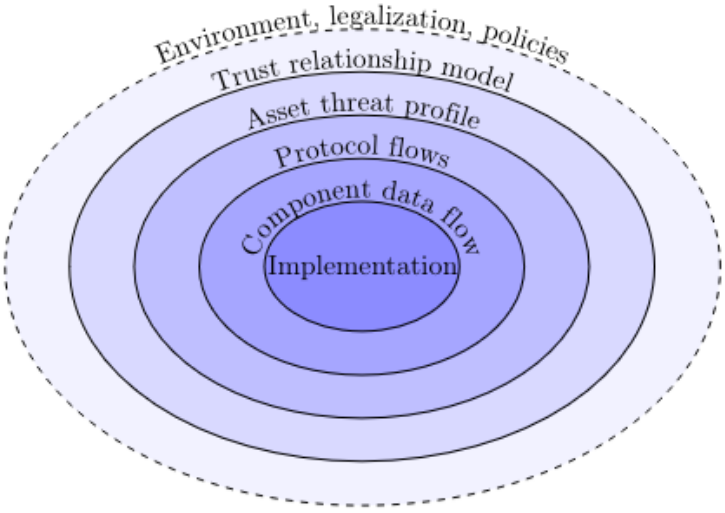

Figure 3. Contextualization of the layered method [18].

The Discovery Phase of IDDIL/ATC methodology [19] consists of 5 activities: identifying the system's assets, defining the attack surface, decomposing the system, identifying any attack vectors, and listing threat actors and their intentions. Oja's [18] layered approach to threat analysis is applied to activity 3, decompile the system, and activity 4, identify possible attack vectors.

2.4 Similar tools

The thesis goal is to propose a design for a SAF. To start the design process, initial requirements must be established (**SRQ1**). Analysing existing security maturity self-assessment tools can help to create requirements and identify gaps for current solutions.

The objective of the following chapter is to compare different types of self-assessment tools related to E-ITS by comparing the solutions' strengths and weaknesses. In order to identify gaps in current solutions, various formats for self-assessment tools were chosen: desktop application, web application, interview format, and working with a file. Because no E-ITS self-assessment web application was found, ENISA's Cybersecurity Maturity Assessment for Small and Medium Enterprises tool was chosen.

2.4.1 Cybersecurity Maturity Assessment for Small and Medium Enterprises

The European Union Agency for Cybersecurity (ENISA) has developed a web-based tool called Cybersecurity Maturity Assessment for Small and Medium Enterprises [21]. The pre-requirements for starting the maturity measuring is to register an account. The user must provide their email, name, position, country, and consent to one of the data processing options:

- agree to share the assessment data with ENISA, including information on the country;
- anonymously share data, without info about the country;
- not agreeing to share data with ENISA, thus the data will be used only in statistics.

The first step of the assessment is to create a business profile by answering seven domain and organisation size questions. The maturity level questionnaire comprises 41 questions ranging from multiple-choice to "Yes/No" options. Based on the input, an improvement action plan is created: a table containing task descriptions, priority, status, comment, and assigned to columns.

Based on the questionnaire results, the organisation's maturity level is Foundation, Advanced, or Expert. ENISA provides three benchmarks, i.e., pie charts illustrating how many organisations in Europe with the same budget, size, or country have the maturity level of Foundation, Advanced, or Expert. Currently, no same-country benchmark data is provided for companies located in Estonia.

2.4.2 Cybsis

Cybsis [22] is a subscription-based (2950€ + tax a year) desktop application for E-ITS implementation. Cybsis gives an overview of implemented and unimplemented security measures and is set up in the user's network. All input and Cybis produced data is encrypted.

The user must create an account, add all organisation assets, and map them to business processes. The user can mark the asset's price, location, if it is active, name, and stock number. Each asset is linked to an elementary threat catalog module attribute and the user needs to mark if it is not implemented, partially implemented, or fully implemented.

It is possible to view the overall state of E-ITS requirements implementation in the organisation. Detailed statistics regarding implementation across modules and business processes are also present. The application includes a protection requirement matrix.

2.4.3 Audit

The E-ITS audit [23] aims to assess whether the information management system and implemented measures are sufficient to protect business-critical assets. The audit consists of a basic audit and an interim audit, preliminary and follow-up audits are performed upon need.

The basic audit [23] is carried out every three years. The auditor bases the audit on organisation's security measures, context, and risk assessment results. The auditor checks objects and measures with previous deficiencies.

The audit [23] is done by interviewing relevant roles, conducting observations and tours, documentation reviews, and operation tests. The final product of the audit consists of two documents- a general conclusion decision and the final report.

The conclusion document [23] does not contain detailed sensitive information, but it highlights the auditor's general assessment of risk management and the implementation of measures. The final report contains the auditor's assessment of the implementation of E-ITS module groups, deviations from the standard, and relevant evidence. However, the

auditor does not present specific recommendations for implementing E-ITS measures in the final report.

2.4.4 F4SLE Excel file

The F4SLE Excel file [13] can be downloaded from the Intenet. The questionnaire consists of around 200 statements the user must answer numerically from 1 to 4. References to the E-ITS Baseline catalog for each attribute is included in the worksheet. After the evaluation is finished, the user can encrypt and mail the results to be included in the benchmark development process.

Based on the questionnaire input, an overall result is displayed as a radar chart in the result sheet. A table of each module four level results is displayed above the chart. Every value of the table is colored either red, orange, yellow, or green to represent the result visually.

2.4.5 Comparison

The comparison of previously introduced self-assessment tools is represented in Table 3. The listed requirements were derived by observing or using the tools. Requirements were selected by identifying each tool's strengths, weaknesses, and similarities with the other solutions.

The conducted tool comparison highlights that all tools include a moderate entry barrier. Before starting the measurement process, prior steps must be taken. Creating an account, setting up a meeting, or configuring the tool requires effort from the user.

Table 3. Similar tools analysis. Symbol '+' marks the presence of a requirement and '-' the absence of it. If the implementation state is unclear, it is marked as 'n/a'.

Requirement	ENISA	Cybsis	Audit	F4SLE Excel
R1. Content is available in multiple languages (e.g., Estonian and English)	-	+	+	+
R2. The user can start measuring the security maturity without performing prior tasks (e.g., asset mapping or tool set up)	+	-	-	+
R3. The measurement tool gives a detailed overview of results (e.g., for each module)	-	+	+	+
R4. No account is needed to start the measuring process	-	-	+	+
R5. The measurement tool provides a benchmark for result comparison	+	+	-	-
R6. Collaboration between different roles is possible for the organisation's measuring process	-	+	+	+
R7. The tool's content is updated regularly (e.g. each year)	n/a	+	+	+
R8. The measurement tool includes the possibility to re-open the results	+	+	+	+
R9. The measurement questionnaire is based on a common standard (e.g., ISO 27001, E-ITS)	-	+	+	+
R10. Customised remediation plan is provided based on results	+	+	+	-
R11. The security measuring is expected to take less than 2 hours to perform	+	-	-	+

The results of the analysis show that no solution encompasses all the derived requirements. The identified gap can be filled by implementing all listed requirements in the security-sensitive SAF. Thus, the comparison attributes will be used as the initial requirements for the SAF.

2.5 Summary

In this chapter, the Estonian Information Security Standard (E-ITS), Framework for Security Level Evaluation (F4SLE), and threat analysis methodology were described. An overview of similar security self-assessment tools is presented. From the comparison results, the first requirements for the design of the SAF are derived (**SRQ1**). The established requirements are utilised for design decisions in Chapter 3.3 and in the proposed design described in Chapter 4.

3 Requirements analysis

In this chapter, additional requirements from the framework manager are collected. Next, the preliminary design development process is described in three iterations. Each iteration shapes the design and proposes limitations and additional criteria for the created framework. Thus, this chapter answers the *SRQ1: What are the requirements to implement the proposed framework design?*

3.1 Scope

This thesis aims to design a framework for repeatedly measuring organisations' information security maturity. The framework should include a component to gather responses to F4SLE and administrative functionality to manage collected data. The designed framework should ensure the collected data secrecy.

3.2 Requirements from F4SLE framework manager

The framework manager of F4SLE discussed the requirements with the thesis author at the start of the project. At the meeting framework manager talk through their expectations for the developed framework. This step included the mapping of needs for the F4SLE tool. The following requirements were expected and grouped by theme.

3.2.1 User functionality

R12. The application must display a questionnaire containing ten 4-level module groups.

R13. The user can evaluate each attribute on a 4-level scale, with the additional options of not answering or marking answer not applicable.

R14. Detailed user input must not leave the user's local disk.

R15. The application must display the results as follows: 4 radar diagrams displaying the results of each level and an overall radar diagram with benchmark overlay.

R16. The application should be security sensitive.

R17. The application should show the benchmarks only when the user has sent their results and additional data.

R18. The user should be able to view the benchmarks without re-sending their results.

R19. The UI should have a low entry barrier.

R20. Submitting the F4SLE results should take as few steps as possible.

R21. The user should be able to save answers to local disk.

R22. The user should be able to continue answering an unfinished version.

R23. The user should be able to quickly move between modules.

R25. The application should visualize the answering progress.

R26. The user should be able to leave text based feedback.

R27. Results and relevant data should be sent to server with explicit consent.

R28. The application should display the benchmark only when the user has submitted the results.

3.2.2 Administrative functionality

R29. The component should allow the selection of benchmarks displayed in the user interface.

R30. The policy maker should be able to access benchmark results, both visual and numeral format.

R31. The policy maker should not be able to access detailed user data.

R32. The framework manager should be able to access user submitted data.

R33. The functionality must automatically calculate benchmarks based on selected types, for example, domain-based or nation-based.

R34. The framework manager should be able to add, modify and delete F4SLE attributes.

R35. The framework manager should be able to modify benchmark calculation and visualization rules.

R36. Benchmark should be calculated based on 5 or more organisations' results belonging to the corresponding group.

3.3 Framework design development process

The thesis is part of a mini project in the CHESS security certification area. Meaning the proposed framework design will be implemented as a proof of concept (PoC) tool and utilized to collect answers to F4SLE. Due to the time restriction regarding the CHESS project, the design and development process took place in parallel.

The framework design process and PoC tool development were conducted in three iterations. In each iteration, decisions regarding the framework were made (marked as D+number) based on gathered requirements and CHESS partners' input.

3.3.1 First iteration

After agreeing on the requirements and thesis outcome, the first iteration included brainstorming for possible design solutions. The thesis author created a graph to identify assets, visualize their exchange between components and identify security needs based on the Confidentiality, Integrity and Availability (CIA) triad [24].

The initial graph design contained a user interface (UI) component, back end component, and administrative component. The output and input produced by the components were visualised with arrows. Lastly, CIA was marked for each asset, illustrating to what extent the asset should be protected.

The E-ITS application module [25] was used to find possible design options for the UI component. The client application submodule included three application types: office software, web browser, and mobile applications.

The thesis author eliminated the mobile application option. The F4SLE evaluation process is expected to take 1 to 2 hours (**R11**). The thesis author finds that working a long period on a small screen could be a barrier (**R19**) and inhibit the responses received.

For deciding between a web-based solution and a desktop application for gathering answers to F4SLE, both options were evaluated against the following attributes:

- **Usability:** first-time application opening process (**R2, R19**); application performance; the comfort of displaying updated questionnaires (**R7**); resource management; Internet requirement.
- **Management:** hard drive usage, management and updates of application (**R2, R19**), backups.
- **Development:** customisability, platforms (Windows, Linux, Mac).

- **Economy:** server management, hosting.
- **Security:** data management (**R14**).

Based on the comparison results, the desktop application approach was preferred (**D1**) as it allows the user to work offline, the user has control over the input data and all data is handled and stored locally. Electron[1] and Vue.js[2] were selected for developing the UI (**D2**) as the technologies allow us to convert a web application to a desktop application. Meaning it is possible to switch between two solutions when required.

In the current thesis, the term security sensitive is used in the following sense: the developed framework must not reveal detailed information about organisations security measures (**R16**). Meaning the evaluations of F4SLE attributes should not leave the user's computer (**R14**), and the data collected must generalize the result (**R27**).

To achieve security sensitivity, only aggregated results will be collected and analyzed by the framework manager (**D3**). Based on the company's responses, an average result is calculated for each module level. The approach is sufficient because it is impossible to know the accurate assessment of each statement, but the result describes the overall implementation state.

The thesis author approached the Republic of Estonia Data Protection Inspectorate to inform them about the planned organisation aggregated security level collection. The author described how the planned activity is compatible with the Public Information Act §35 point 9 [6]. The Data Protection Inspectorate did not prohibit the activity. However, the discussion regarding the accordance continues.

As accounts and authorisation functionality will not be included for the user interface (**R4**), it raises the question of how to manage multiple responses for one organisation. The options considered as a solution were distributing tokens, asking users to encrypt and send answers via email, or implementing Estonian ID card encryption to the desktop application.

As the pilot involves Czech Republic respondents, Estonian ID card encryption was not decided on. Asking users to save their results, encrypt them, open their mail server, and send the answers to the correct e-mail address requires effort from the user and may result as a barrier. Thus, this may limit the number of responses.

Tokens could be implemented in various ways: distributed to the participants before starting the answering process via separate communication channel, providing tokens only

[1]Electron (url: https://www.electronjs.org/ Accessed 02.05.2023)
[2]Vue.js (url: https://vuejs.org/ Accessed 02.05.2023)

to responded users, generating tokens with each desktop application download, etc.

When the user receives tokens separate from the tool, they need to remember or store them in a safe location and correctly type them into the input field, hence leaving space for errors. Thus the automatisation of the token distribution and submission process without the user's partaking was opted for (**D4**).

The development of a proof of concept tool called MASS (Measurement Application for Self-assessing Security) was started at the end of the first iteration. Based on the conducted analysis and decisions the framework design and PoC should include:

- a desktop application for evaluating organisations' security maturity,
- a back-end for handling requests,
- a database for storing unique tokens;
- a server for hosting the back-end and storing the downloadable desktop application.

3.3.2 Second iteration

The initial plan was to host the desktop application and back end on the CHESS project's Czech team's server. After the first meeting, it became clear that the partners had not the required resources and could not set up the server. Thus, the application and other related data should be available for download from the University of Tartu server (**D5**).

The discussion with the Czech party revealed that redirecting Czech users to the Estonian website to download the desktop app might present mistrust in the usage of the application and limit the response amount.

Furthermore, the participating Estonian organisations must implement E-ITS and meet APP requirements. APP.6.M3 states [25] that all downloadable software has to be validated against APP.6.M2 requirements list. Therefore, the user has to get permission from the IT department to download the tool, and it might present as an entry barrier. An interview with a Czech expert revealed that the same constraint exists for Czech organisations.

Considering the gathered input, the thesis author decided to implement a web-based solution for the user interface (**D6**).

In the second iteration, the metadata gathered from participants was agreed on (**D7**). The collected metadata is used to perform data analysis centrally and to remove duplicate or false results. Predefined options were added (e.g., domain list, role list, standard list)

to simplify data analysis. The full list of collected metadata is described in Table 1 in Appendix 5.

3.3.3 Third iteration

The discussion regarding administrative interface requirements and needs revealed that the final data processing shape and visualization are unknown. This means that the administrative part of the framework should be as flexible as possible. A flexible approach is not possible when the administrative component is a web-based or desktop application with pre-defined functionality.

The framework manager currently uses a Jupyter Notebook file to analyze and visualize data. The manager stated that the current approach allows to adjust the benchmark calculation process upon need. The thesis author decided to utilize the already existing Jupyter Notebook file in the administrative functionality (**D8**) and build the administrative component around the file.

Therefore, a server-based solution was opted for and built upon (**D9**). By designing the administrative functionality as a server-based solution, we can narrow the scope for potential threats by using continuously patched, tested, and developed built-in functionality.

The Jupyter Notebook file can be configured to output data (e.g., graphs, benchmark numeral results) to pre-defined locations. In addition, the server allows to define roles and configure access to files hosted on the server based on the group (**R30, R31, R32, R34, R35**). For example, the framework manager can read, write and execute a Jupyther Notebook file for benchmark calculations (**R35**). However, a policymaker can only access the benchmark results file (**R30**).

Based on previous iterations feedback, discoveries and analysis the final design for the framework was developed and the PoC MASS (Measurement Application for Self-assessing Security) was finalized. The pre-test of MASS and its launch in Estonia and Czech Republic were part of the third iteration. With the gathered input, the framework manager began the initial data analysis.

3.4 Summary

This chapter derived requirements for the developed security sensitive framework, thus answering the **SRQ1**. Next, three iterations of the framework design development were conducted. By the end of the third iteration, a total of nine decisions were made regarding the framework's possible structure based on identified limitations and needs of the involved parties.

The requirements and decisions made in this chapter will be utilized to propose a security sensitive SAF design in the next chapter (**PRQ**).

4 Security sensitive Self-Assessment Framework (SAF)

In this chapter, the primary research question *How should the security sensitive self-assessment framework design look like?* is answered. Based on Chapter 3 analysis, derived requirements, and identified limitations a framework design is proposed. In addition, a limited proof of concept framework is developed.

4.1 Actors

Based on the requirements defined in Chapter 2.4 and Chapter 3.2 and decisions **D6, D9**, five main roles were identified in the workflow:

Framework manager (FM) One or few per instance.
Creates and manages the questionnaire and participant data.

Organisation representative (OrgR) One to thousands per instance.
User, who fills in the questionnaire on behalf of the organisation. Usually security or IT manager.

Software provider (SP) One or few per instance.
Develops and maintains framework services.

Server administrator (SA) One or few per instance.
Maintains the server and user groups.

Policy maker (PM) One or few per instance.
Analyses collected aggregated data.

4.2 Proposed architecture

Fig. 4 illustrates the proposed design for a SAF. The self-assessment tool is a public web application hosted on a server. All calculations are made on the client side; only aggregated results and metadata are sent to the back end with the user's explicit consent.

The back end handles requests, generates and manages tokens, and stores user submissions on the file server. Access to the file server is granted based on pre-defined roles. The back and front end is developed, maintained, and deployed from a code repository.

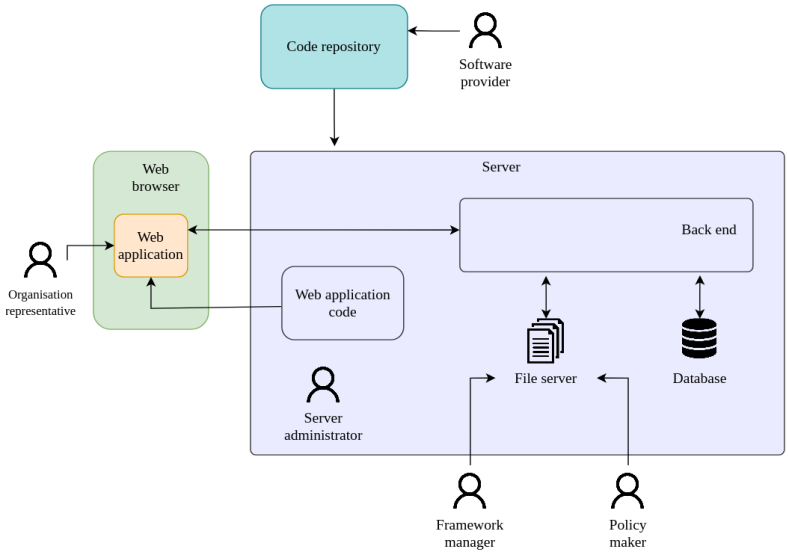

Figure 4. Self-Assessment Framework (SAF) architecture

4.3 Administrative functionality

4.3.1 User permissions in server

Fig. 5 illustrates the user permissions in the administrative server. A valid user can connect to the server using University of Tartu virtual private network (VPN) connection and Secure Shell (SSH) protocol (**D5**). Based on the authorisation request, each valid user is granted corresponding privileges, either SA, PM or FM.

The FM can read, write and execute all data related to the framework (**R32, R34, R35**). The PM can read all data related to the framework except organisation representative data and aggregated results (**R30, R31**). A Jupiter Notebook (**D8**) will be automatically run to generate benchmark calculations and visualisation (**R33**). Results will be stored in corresponding folders.

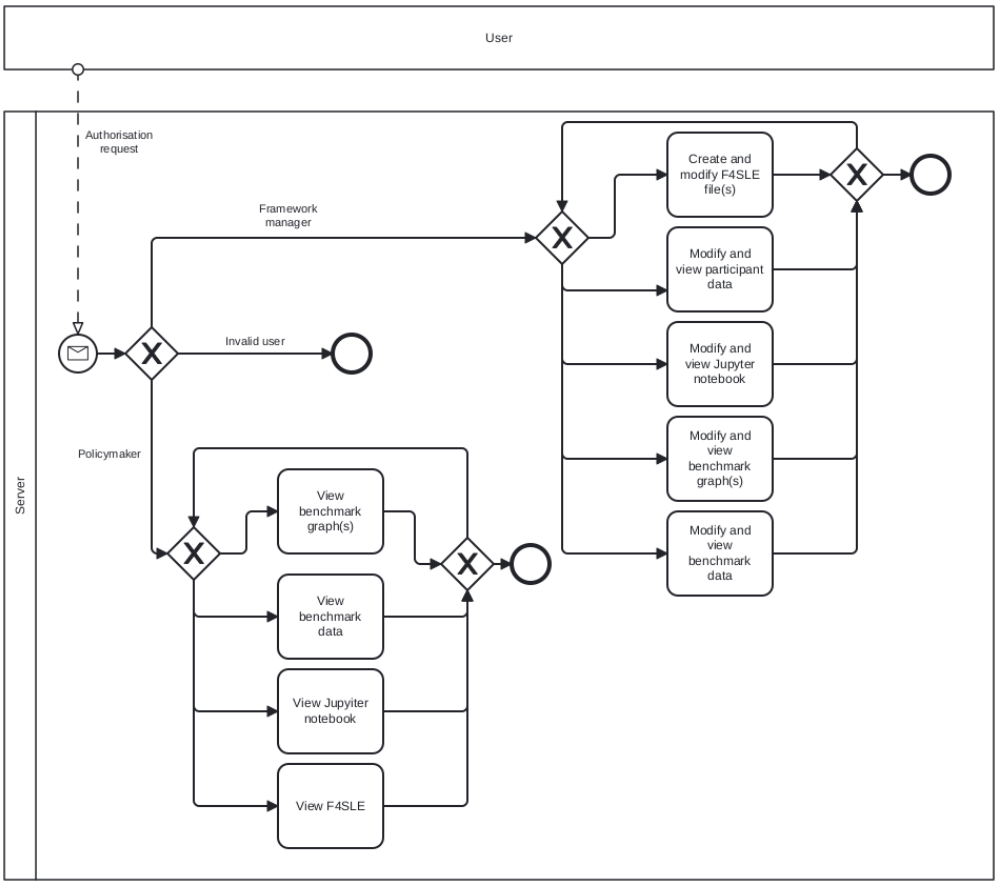

Figure 5. User permissions in the server

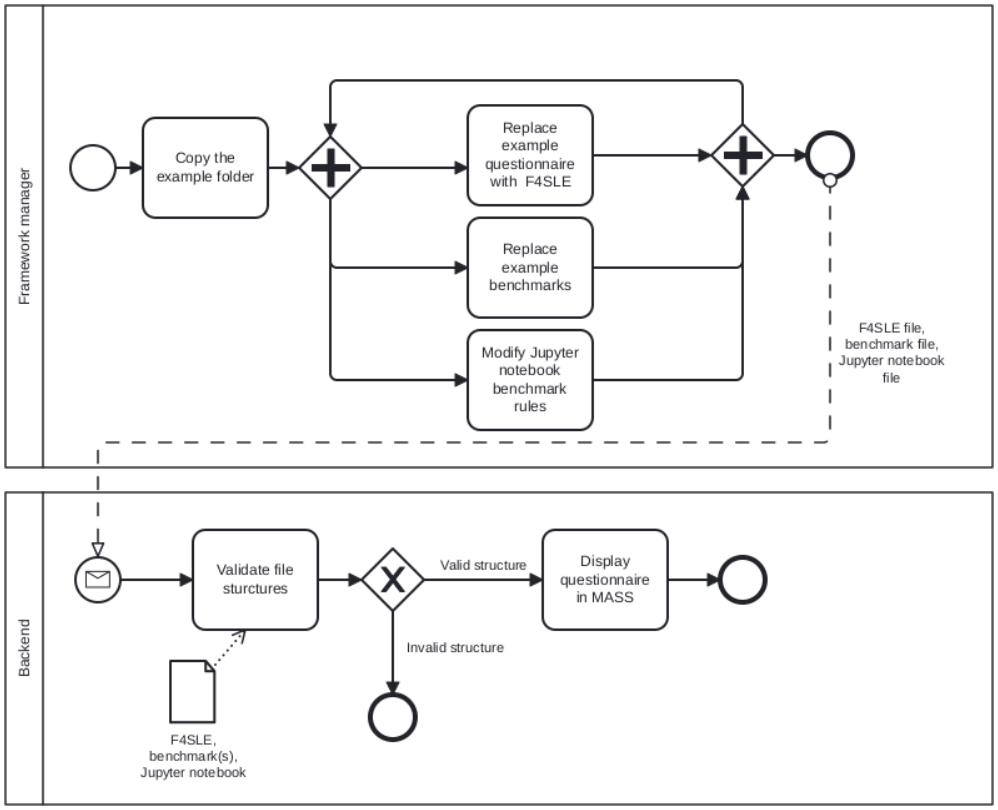

Figure 6. Questionnaire creation flow

4.3.2 Questionnaire creation

The questionnaire creation process is shown in Fig. 6. The framework manager can copy, edit and rename an example folder with the required questionnaire setup (**R34**). The example folder consists of a questionnaire JSON file, an initial benchmark file, a Jupyter Notebook file, and directories for collected organisation results, benchmark calculation results and graphs. When the back end detects a new questionnaire folder, the structure of all files is validated. In case of an error, the FM needs to resolve the issue and go through the whole creation process again. If no errors occur, the new version is displayed in MASS.

If less than 5 entries to the questionnaire have been received, it is not possible to calculate a benchmark (**R36**). The limit is set by the FM due to privacy considerations. As the benchmark is the average of responded organisations' results, a small sample might reveal aggregated results. For example, an attacker knows that orgnasiation Alpha

has answered to F4SLE, and the overall benchmark value is low for the DER module. When the benchmark is calculated for less than 5 orgnaisations, we can assume that Alpha's DER module is weak too. This kind of information can help an attacker to craft a targeted attack.

The initial benchmark file allows the definition of results that are displayed to the user (**R29**). For example, the framework manager can add the last year's overall benchmark and health domain benchmark to be displayed in the UI.

4.3.3 Data collection

Fig. 7 displays the data flow when an organisation representative sends the aggregated results and additional data to the server. Firstly the back end validates the JSON input received from MASS. When the input is valid, the back end checks the questionnaire version, locates the corresponding folder, generates a unique file name, and stores the user-submitted data.

When the representative-submitted data is successfully stored on the server, a unique token (**D4**) is sent to the user. This token proves that the user has submitted evaluation results and can view benchmarks without re-submitting data. The token is stored in a downloadable result file and will be derived from the file when the organisation re-visits results.

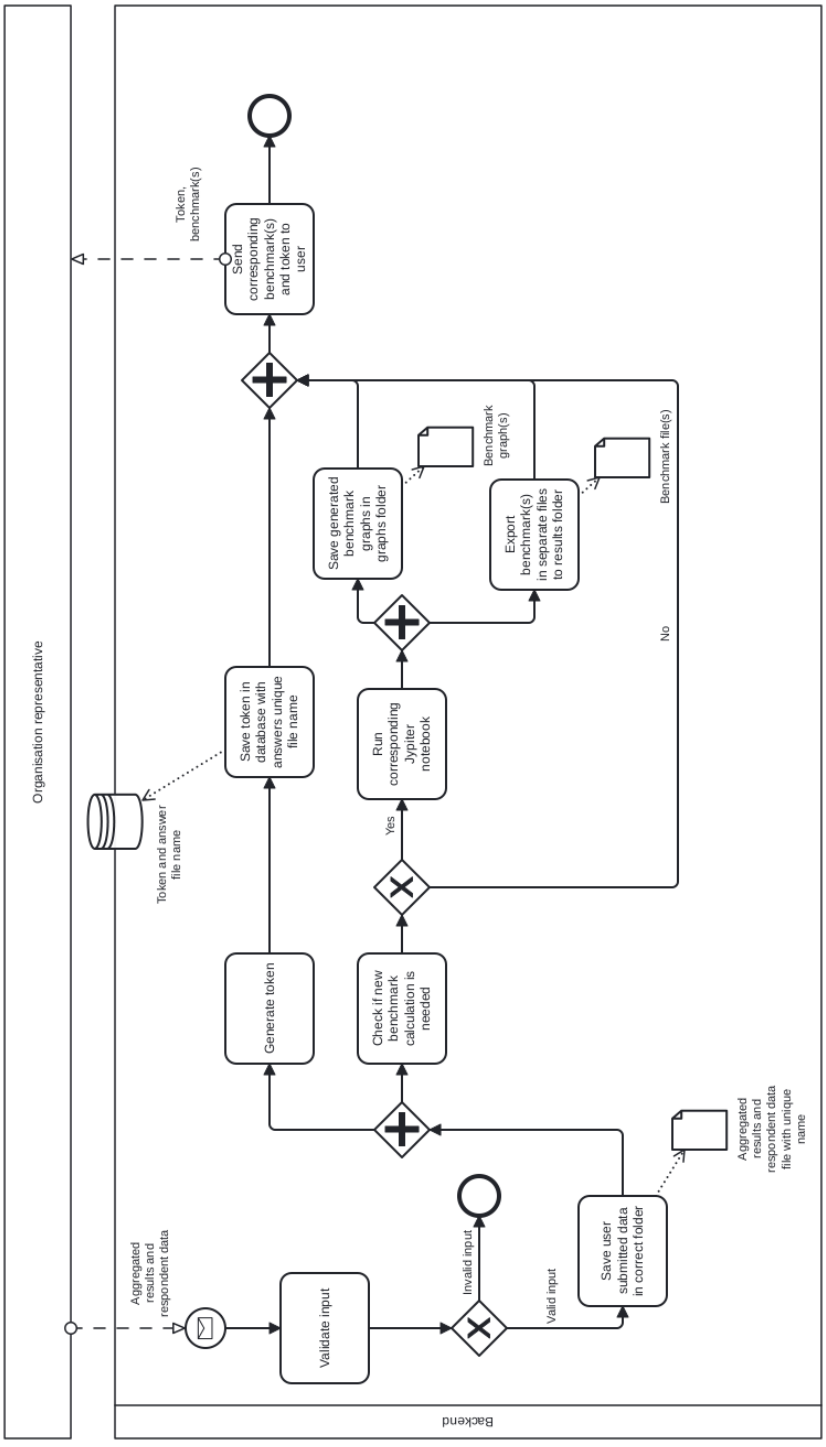

Figure 7. Organisation representative data collection flow

4.4 Measurement Application for Self-assessing Security

A representative can evaluate their organisation's current information security state using the web-based application (**D6**) called MASS. The application presents the framework manager-created questionnaire file to the user. The answer collection process is described in Figure 8.

The organisation representative can start the information security maturity evaluation process without performing prior tasks (**R2, R4, R19**). MASS can be divided into five main views: landing page, options page, questionnaire, results, and data collection view. All views are available in both Estonian and in English (**R1**). The user interface design is simple and the color pallet is inspired by the E-ITS web-page[3].

4.4.1 Landing page

Fig.2 in Appendix 4 illustrates the landing page of MASS. The view includes an introduction of the tool and guidelines. In addition, principles of data processing are introduced to the user.

The main principles are:

1. Detailed answers exist only on the respondent's computer;
2. Aggregated number values of organisation's maturity will be sent to the server only with explicit consent from the user;
3. It is possible to review the data before sending the aggregated number values to the server;
4. The benchmark will be provided when the the participant successfully transmits the aggregated number values to the server.

The main principles of data processing are:

1. The e-mail address field is optional and only used to inform participants about a new questionnaire version or new benchmark data;
2. All submitted data is analysed and presented without the names of the participating organisations;

[3]E-ITS (url: https://eits.ria.ee/ Accessed 13.04.2023)

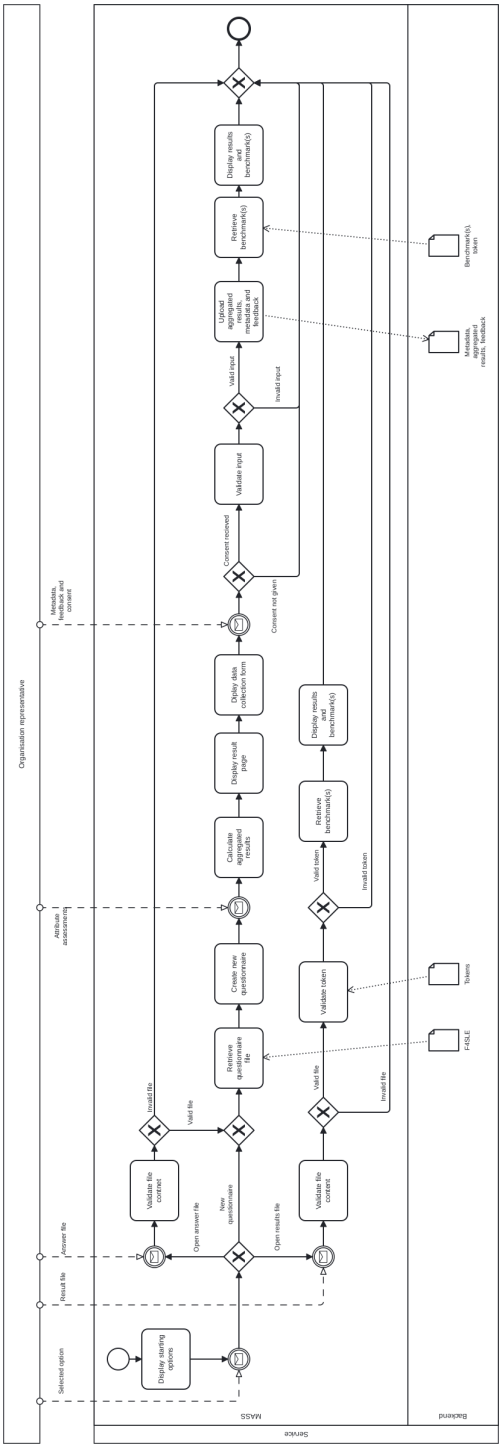

Figure 8. Organisation security maturity evaluation data collection flow

3. All published benchmarks (domain based, size of institution, etc.) are calculated from at least five institutions belonging the the corresponding group;

4. In cooperation with Estonian National Cyber Security Centre (RIA, NCSC-EE), researchers will use the data for research and develop a domain-based and general benchmark at the national level.

4.4.2 Options page

In the options page shown in Fig.3 in Appendix 4, the user is presented with three choices for actions related to F4SLE.

Start a new questionnaire. The option creates a new blank questionnaire. To open the newest version of F4SLE a request is forwarded to the back end using the Internet. The user is notified of the action by a pop-up and they can continue or decline it.

Open an existing answer file from local disk. This functionality allows users to open a previously saved answer file from the local disk (**R22**). The option can be used to continue answering uncompleted responses, examine previous inputs, and share the answering obligation within the organisation or between roles (**R6**).

The user is notified about the back-end request to obtain the newest version of F4SLE by a pop-up, and they can continue or decline it. Answer values are ignored if the statement is no longer part of F4SLE.

Open an existing result file from local disk. This option can be selected to open the result view for completed questionnaires (**R8**). When the uploaded data file contains a unique token, the corresponding benchmark will be requested from the server. The user is notified of the action by a pop-up and they can continue or decline it.

4.4.3 Questionnaire

An example questionnaire view is shown on Fig.9. The view contains two main containers: the navigation container on the right and the questionnaire container beside it. The navigation container can be used to move between modules by clicking on the desired module abbreviation (**R23**). A check mark will appear before the module abbreviation if all questions belonging to that module are answered (**R25**). A count of evaluated statements is displayed at the top of the module list (**R25**).

The questionnaire container displays one module at the time (**R12**). The module name, abbreviation, and description are displayed at the container's beginning. All questions belonging to the module are numbered and listed below.

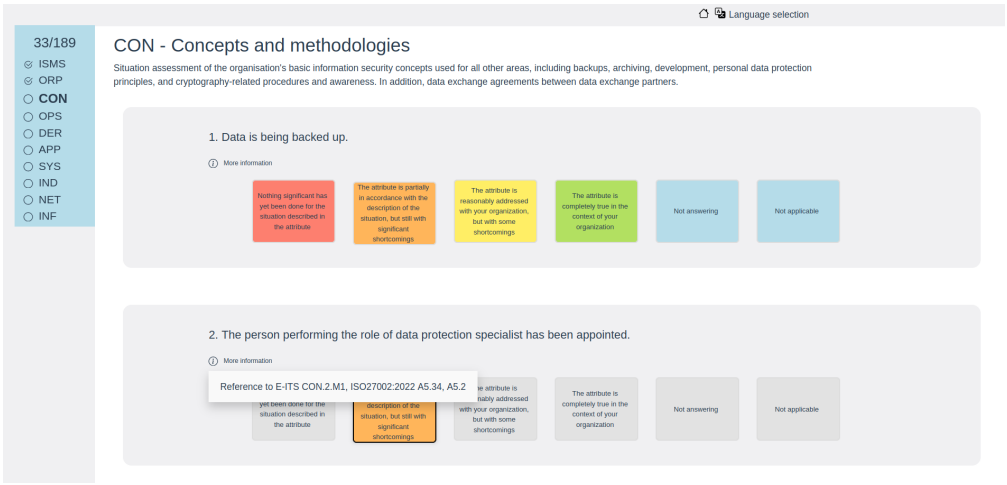

Figure 9. Screenshot of MASS questionnaire page [26]

The user can evaluate all statements in a four level scale (**R13**), additional option "Not answering" and "Not applicable" are provided in order to skip evaluation. Each statement can be evaluated by clicking on the suitable answer box. Not selected answer options will turn gray. Additional data will be displayed when hovering over the information icon or "More information" text. To save the current answering state the user should click on the "Home" button and download the answer file (**R21**).

At the end of the container, navigation buttons "Next" and "Back" are displayed. After all statements are evaluated, the "End" button is enabled, and moving to the result screen is possible.

4.4.4 Results

Fig. 4 in Appendix 4 displays the result page first half. All answers to F4SLE and calculated results can be saved by clicking on the download button on the right (**R21**). The radar diagram portrays the organisations' aggregated results for each domain and the recommended goal (**R5, R15**). When the organisation's representative has sent the aggregated results to the back end through the data collection form, benchmarks will be added to the diagram (**R17**).

Additional information is added to the screen to help the user interpret the results. For example, next to the graph is the legend that defines the result value meaning. Each module abbreviation and definition are included bellow the graph.

The second half of the result screen shown in Fig. 5 in Appendix 4 includes a detailed

view of each level result (**R3, R15**). The four radar diagrams correlate to the four levels that each domain contains. Definitions of each level are displayed below the plots.

All result calculations are made on the client side (**R14, R16**). Based on the user's evaluations of each statement, the aggregated results are calculated with the following steps. Firstly, each answer is processed based on the rules:

- Value red or "Nothing significant has yet been done for the situation described in the attribute" is interpreted as 0.
- Value orange or "The attribute is partially in accordance with the description of the situation, but still with significant shortcomings" is interpreted as 1.
- Value yellow or "The attribute is reasonably addressed with your organization, but with some shortcomings" is interpreted as 2.
- Value green or "The attribute is completely true in the context of your organization" is interpreted as 3.
- Value "Not answering" is interpreted as null.
- Value "Not applicable" is interpreted as null.

Next, four benchmarks are calculated for each module level: the sum of level answers with values 0 to 3 is divided by the count of level answers with values 0 to 3. When the sum of answers equals null, the value will not be displayed on the radar diagram.

Finally, the overall result is calculated based on four level values. The sum of each module's levels is calculated and divided by the count of levels with the value not null.

4.4.5 Data collection form

To view relevant benchmarks, the user needs to send the aggregated results and metadata (**D3,D7**) using the data collection form shown in Fig. 6 in Appendix 4.

As a control mechanism, viewing the sent data and validating its trueness is possible by clicking on the "Preview of information sent to database" text or info icon. The user is displayed the exact data in JSON format, that is forwarded to the server. This functionality is illustrated in Fig. 7 in Appendix 4. Only with the explicit consent (**R14, 27**) from the user, aggregated results (**D3, R16**) and metadata (**D7**) are sent to the server.

When the aggregated data is successfully received by the server, a unique token is returned. The token is automatically added to the downloadable results file and is used when requesting benchmarks (**D4**) in the results view.

4.5 Proof of concept

In order to validate the created framework design and collect answers to F4SLE, a limited version of the framework was developed. The PoC included realisation of MASS and administrative functionality. All requirements, except 4 were implemented in the limited PoC due to time limitation: automatic data processing (**R33**), the policy maker role (**R30, R31**), and providing a customized remediation plan (**R10**) are considered out of scope for the PoC.

The thesis author was responsible for developing the low barrier-of-entry user interface for F4SLE. Server and database configuration was implemented by the server administrator, programming of the back-end by the software provider. Jupyter Notebook file for the benchmark calculation was provided by the framework manager.

Open-source technology was preferred when developing the PoC. JavaScript framework Vue[4] version 3.2.45 was used to develop the user interface (**D2**). The back-end was written in Python[5] version 3.10 and hosted with Flask[6] version 2.2.2.

The MASS user interface is available at `https://mass.cloud.ut.ee/massui` [26]. The application and back end are hoster on the University of Tartu server. All development and deployment is managed by using the University of Tartu GitLab[7] code repository.

4.6 Summary

In this chapter, the primary research question *How should the security sensitive self-assessment framework design look like?* was answered. The proposed design includes a security sensitive web application for self-assessing security, and all back end and administrative functionality is hosted on the server. Each identified requirement (Chapter 3) is mapped to the design.

A limited version proof of concept of the framework is introduced at the end of the chapter. The implemented solution is utilized in the next chapter to validate the developed design using a pilot group.

[4]Vue.js (url: https://vuejs.org/ Accessed 13.04.2023)
[5]Python (url: https://www.python.org/ Accessed 13.04.2023)
[6]Flask (url: https://palletsprojects.com/p/flask/ Accessed 13.04.2023)
[7]GitLab (url: https://gitlab.ut.ee Accessed 09.08.2023)

5 Validation

The following chapter seeks to validate the developed self-assessment framework proposed in Chapter 4. To achieve this goal a two-part validation is performed. Firstly, the *SRQ2: What is the security posture of the proposed framework design?* will be addressed by conducting a threat analysis. Secondly, the implemented proof of concept framework is validated by a pilot group. Selected trusted participants will use MASS to measure their organisation's security maturity and assess their user experience.

5.1 Threat analysis methodology

A. Shostack [17] argues that there is no one right way to model threats and proposes a general 4-step framework that can be customized based on time, experience, tools, and other resources. Firstly, it is vital to understand what is being built: describe the system, boundaries, assets, and flows. The second step is to find possible threats. Threats can be identified by asking what can go wrong and where error-prone places can be. A systematic approach to identifying threats can be achieved using the STRIDE mnemonic (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privileged) described in Chapter 2.3 for conducting threat profiles.

The third step consists of addressing the found threats. Each threat should be assigned an action [16, 17, 27]:

- mitigating the threat by implementing countermeasures to reduce the probability of the risk or lower the residual risk;
- eliminating the threat by removing the feature, asset, or component so that it no longer poses a security threat;
- transferring the threat or threat outcome to a third party;
- accepting the threat.

Lastly, findings should be validated. Validation includes a review of all previously conducted steps. Activities include evaluating current work, implementing improvements, and building tests for found problems [17].

The threat analysis conducted in the following chapter will follow the general 4-step threat modelling framework and combine Oja's [18] and Muckin et al. [19] proposed methods (discussed in Chapter 2.3) for step 1 and 2.

5.2 Assets

In order to start conducting a threat analysis, it is necessary understand the developed framework. This can be achieved by identifying and describing assets: anything that creates value or needs to be protected [28]. Assets can be, for example, a laptop, system administrator role, API endpoint, software, or programming competence.

The goal of an attacker is to affect the availability, integrity or confidentiality of assets and thus cause damage to the system or disturb the business process. Mapping all assets creates an overview of what needs to be protected and helps to identify possible threats unique to the asset. Based on the developed framework design (proposed in Chapter 4), the following assets were identified and listed in Table 4.

Table 4. Identified assets.

Asset	Description
Questionnaire metadata	Benchmark address - the endpoint where the benchmarks are requested from. Answers' address -the endpoint where the user submitted data is sent to.
Questionnaire	F4SLE - the security maturity evaluation questionnaire. Version number - automatically generated with each new F4SLE version and used to track corresponding benchmarks and results.
Organisation representative data	Participant metadata - additional information asked form user (e.g. organisation domain, implemented policies, feedback).
Aggregated results	Organisation's 10 x 4 aggregated results of each module group.
Detailed F4SLE answers	User provided assessment for each attribute. Used by MASS to calculate aggregated results. This data is not sent to server nor stored in browser cache.
Benchmark	Automatically calculated benchmark for result comparison.

Token	Automatically generated token for users to access the benchmark.
Server	Hosting the web-application MASS and back-end functionality, storing questionnaire data.
MASS	Self-assessment tool for F4SLE.
Back end	Back-end functionality running on the server.
Jupyter notebook	File for benchmark calculation and data analysis.
Framework manager role	Server role that can create and modify F4SLE files, participant data, Jupyter Notebook file, benchmark graph(s) and data.
Policy maker role	Server role that can view F4SLE files, Jupyter Notebook file, benchmark graph(s) and data.
Software provider role	Develops the web-application functionality.
Server administrator role	Develops the back-end functionality, maintains the server.
Organisation representative	Individual who uses MASS to provide answers to F4SLE questionnaire.
GitLab code repository	External development platform used for storing code and deploying the web-application.

5.3 Attack surface

The attack surface defines system and trust boundaries and helps to determine the scope of the threat analysis [19]. Previously identified assets are mapped to the attack surface illustrated in Fig. 10.

The data stored in the server (questionnaire data, OrgR data, aggregated answer files, benchmarks and graphs, Jupyter notebooks, tokens, back end code, front end code) is grouped under one asset to simplify the figure. In addition, 2 components are added to

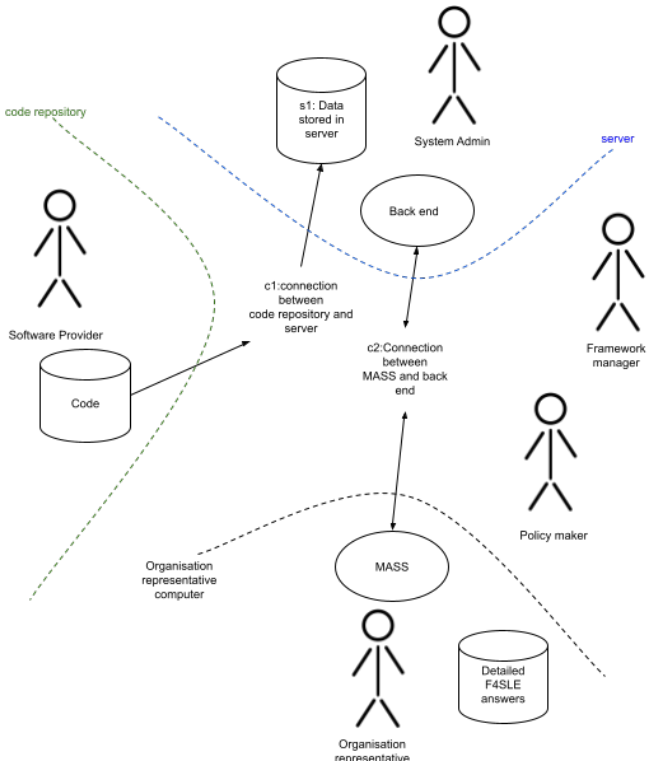

Figure 10. Attack surface

the asset list:

c1 The MASS and/or back end code update.

c2 The connection between MASS and back end.

Figure 10 helps to visually represent identified framework assets and the communication between them. We can utilise the figure to identify possible threats. For example, the asset c1: connection between code repository and server can be attacked to disrupt the back end update.

The attack surface can be used to identify boundaries between the components. For example, both the Policy Maker and Framework Manager are separate entities and must be authorized to access the server.

5.4 Assets threat profiles

Asset threat profiles can be used to concentrate possible threat information in one place. These profiles can be used to understand how an attacker can harm the current system, and they can be referenced during the system development lifecycle. Threat profiles help identify possible attack vectors, surfaces, and actors.

Muckin et. al. [19] created threat profile template was modified to adjust to the scope of the threat analysis. Thus, vulnerabilities, controls and resultant conditions are not included in the profile. Asset threat profiles were created to analyse potential threats based on the STRIDE model (described in Chapter 2.3).

The possible threats were conducted based on previous experience, the OWASP top 10 most critical security risks to web applications list [29], and by brainstorming. An example threat profile can be found in Table 5 illustrating asset participant data. The extensive asset threat profile list can be found in Appendix 5.

Table 5. Threat profile: Organisation representative data

	Description
Asset	**Organisation representative data**
Threat types	STI
Attack surface	MASS Server Code repository Administrative operations Network
Attack vectors	Development flaw: server or MASS leaks data, configuration flaw Human error Social engineering Modified participant answers with MitM Manipulating server, backend or MASS Sending aggregated results as another OrgR

Threat actors	Malicious or uncaring SA/FM/PM/SP/OrgR APT or script kiddie

5.5 Protocol flows

Tarmo Oja [18] proposes a fishbone/mind map approach based on the Ishikawa cause-effect diagram [30] to map protocols that create, transform or transport trusted assets. The graphical representation can be used to update or review flows in the threat analysis process.

The protocol flow graphs are read from left to right, with the outcome marked as the rightmost box element. Ovals located above and below the center arrow are elements that contribute to the successful outcome. Each element can have multiple contributing inputs. An example protocol flow can be seen in Table 11.

input The red color marks a critical input that disrupts the business flow when confidentiality or integrity is lost.

| Blue lines mark system or trust boundaries.

input Underline item indicates that human interaction is needed.

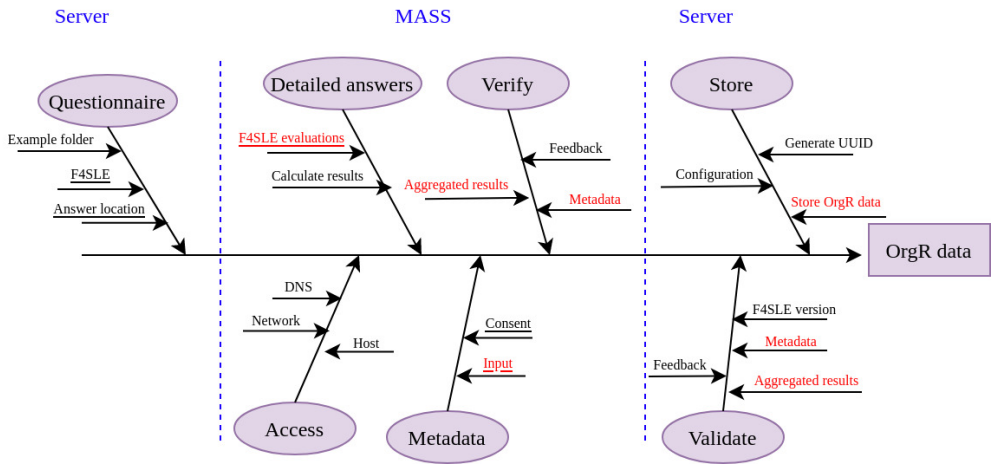

Figure 11. Protocol flow: Organisation representative data

Figure 11 illustrates the protocol flow between the server and MASS. Firstly, the FM creates a questionnaire and data storage location. Then an organisation representative can access MASS and evaluate the latest F4SLE version. When the representative provides additional metadata and consent, the data is transported to the server. Successfully verified OrgR data is stored for benchmark calculations.

5.6 Threat analysis

The analysis performed in previous chapters can be utilized as preparation to identify potential threats. Assets, attack surface, and asset threat profiles help to identify the designed system and its components. Conducted protocol flows map the positive scenarios in the framework.

The proposed framework's security posture (**SRQ2**) can be evaluated by determining potential threats to the system and its components and asking "what could go wrong" in the identified positive scenarios.

Based on the potential weaknesses identifying method developed by Oja [18] a threat analysis for the designed security-sensitive self-assessment framework was conducted. The threat analysis is added in Appendix 8 Table 16.

Table 6 show an example finding from the threat analysis table, referenced in Appendix 8. Each potential threat is presented as follows:

ID - an identifier to distinguish between different threats;

Target - asset or attack surface that is targeted in the threat;

STRIDE - threat type based on STRIDE category (described in Chapter 2.3);

Threat - general description of the possible threat;

Action- suggested treatment measures.

For example, Table 6 shows that actions I01-A01 and I01-A02 are implemented for threat I01. However, actions I01-A03, I01-A04, and I01-A05 are currently not implemented. Thus, the potential threat of MASS leaking data due to development flaw exists.

Table 6. Example finding from the threat report in Appendix 8. Actions marked in bold are measures that are already implemented in the limited proof-of-concept.

ID	Target	STRIDE	Threat	Action
I01	MASS	I	MASS leaks data (development flaw)	**I01-A01:** User answers are not logged or stored in browser cache. **I01-A02:** Minimal input form user is collected and sent to server. I01-A03: All F4SLE modules must include at least two attributes to avoid revealing an attribute's exact evaluation. I01-A04: Secure by design, secure by default and secure by deployment approach should be implemented throughout the framework. I01-A05: MASS is OWASP ASVS 4.03 level 2 compliant [31]

5.6.1 Identified threats

Table 7 gives an categorized overview of the threat analysis report findings. Altogether 37 possible threats were identified, 124 actions were mapped for possible threats, and 66 of them are already implemented for the limited proof of concept solution. It is important to note that one action can appear under multiple threats.

Table 7. Threat category summary.

Category	Total	Actions	Implemented actions
Spoofing	5	15	12
Tampering	8	39	17
Repudiation	4	4	3
Information disclosure	13	38	22
Denial of service	5	17	6
Elevation of privilege	2	11	6

Currently, hardening the system was not carried out in the limited framework implementation. In addition, there was no emphasis on implementing security measures for the back end (e.g., request content validation). In the future, all actions should be implemented to mitigate possible threats. For example, the MASS application should aim to be OWASP ASVS 4.0.3 level 2 [31] compliant, and the server should limit role privileges.

5.7 Pilot in Estonia and Czech Republic

A pilot test was conducted to validate the developed framework and MASS. The first stage was a pre-test to collect user feedback on the application and identify bugs or other issues. The pre-test group consisted of two testers: the IT Manager of a Municipality with over ten years of work experience and the Chief Information Security Officer of a finance organisation with over ten years of work experience.

Most identified issues and feedback in the pre-test phase were corrected or implemented. For example, several typos in F4SLE and web application text were fixed, E-ITS was added to the implemented standards list, and navigation functionality (**R23**) was improved on.

The second stage of the pilot test included the distribution of MASS. The thesis supervisors and the thesis author sent invites to select trusted parties via email. The participation invite highlighted the benefits of the tool and MASS usage guidelines.

Participants were asked to provide feedback about the Measurement Application for Self-assessing Security (MASS) user experience. The feedback was collected using an anonymized form created with CryptPad[8]. Completion of the form took 2-5 minutes and the questionnaire is included in Appendix 3. The survey's goal was to get input on how the participants perceived their information secrecy using MASS.

5.7.1 Pilot results

The pilot took place 13.03-12.06.2023, 30 organisations have participated in the Estonian pilot test group and 4 in the Czech Republic test group. For the result analysis the data is viewed as a whole and not on a national basis. Table 8 shows the participated organisation grouped by domain.

[8]CryptPad (url: https://cryptpad.fr/ Accessed 10.04.2023)

Table 8. Participating organisations

Domain	1...30 workplaces	31...100 workplaces	101...300 workplaces	301...1000 workplaces	1001... workplaces
Education	0	3	2	2	3
Government office	0	1	0	0	1
Healthcare	0	2	1	2	0
ICT	1	0	1	0	0
Municipality	1	4	4	1	0
Non-profit	1	1	0	0	0
Other private sector	0	1	0	0	0
Other	0	0	0	2	0

Fig.12 shows the benchmarks for municipality, education, and healthcare domains. The domain benchmarks could be calculated because five or more organisations belonging to the corresponding group submitted the aggregated results (**R36**). The green sector represents a result guideline that organizations should aim for. Overall all three domains have similar results across all 10 E-ITS modules. The benchmark was calculated and visualised using the administrative functionality and Jupyter Notebook (validation of **D7, D8, D9**). More detailed results are described in the paper "Security level evaluation with F4SLE" [4].

Out of the 30 Estonian pilot participants, six answered to the feedback form. All six individual stated that assessing their organisation's security maturity with MASS caused interest, excitement, and satisfaction. Five participants evaluated that they felt the application is rather secure, and one found it very secure.

When asked about the preferred tool for security maturity assessment, five users voted for a web-based application (validation of **D6**), one for a desktop application, and non of them voted for working with an excel file or conducting the assessment as an

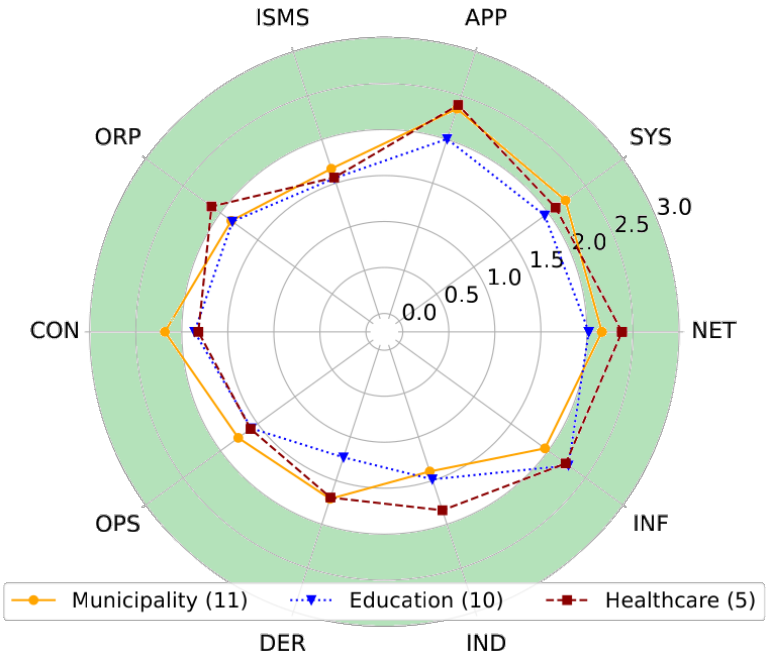

Figure 12. 2023 domain benchmark results

interview/audit.

The participants were asked to evaluate which application components ensured the transparency of data analysis. The results are illustrated in Fig.13. Users found that the preview of transmitted data, the option to opt out of data transmission, and the data collection form component ensured transparency in MASS application.

One participant added that the transmitted data preview component allowed them to verify the exact type and format of the collected data. They added that seeing the aggregated data made them want to share the results, and they felt in control of their detailed answers (validation of **D3**).

5.8 Summary

In this chapter, the proposed design of a security sensitive self-assessment framework was validated. Firstly, the security posture of the described design in Chapter 4 is evaluated by conducting a threat analysis. The threat analysis consists of possible threats and actions regarding treatment measures. The identified actions can be utilized as additional requirements for the proposed framework (**SRQ1**).

A pilot test was conducted for the limited framework implementation. Input regarding

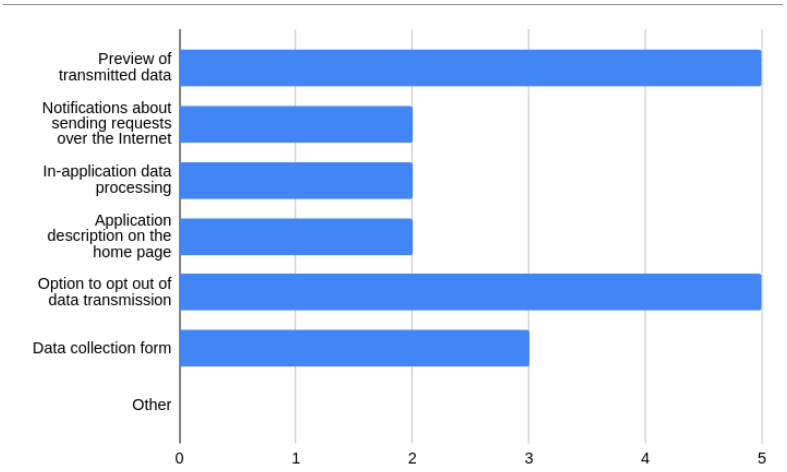

Figure 13. Results of data analysis transparency evaluation

the tool design decisions was collected, resulting in decisions **D3**, **D6**, **D8**, and **D9** being validated.

The threat analysis proved that the framework's security posture is adequate when all suggested actions are implemented. In addition, the pilot results validated the stated requirements and decisions described in Chapter 3.

6 Conclusion

This thesis provides a design for a security sensitive self-assessment framework based on F4SLE and a limited proof of concept implementation of the proposed framework. To validate the developed solution, a pilot test in Estonian and Czech Republic was conducted and a threat analysis was carried out.

Answers to research questions

PRQ: How should the security sensitive self-assessment framework design look like?

Answer: Chapter 4 provides an overview of the developed self-assessment framework. Decisions and limitations regarding the proposed design are described in Chapter 3.3.

SRQ1: What are the requirements to implement the proposed framework design?

Answer: In Chapter 2.4 gaps in similar tools were identified, in Chapter 3 requirements from the framework manager and relevant parties are derived. Chapter 5.7 describes the selected requirements validation.

SRQ2: What is the security posture of the proposed framework design?

Answer: In Chapter 5.6 a threat analysis was conducted based the designed framework. Additional requirements for the framework were mapped as a result (**SRQ1**).

6.1 Limitations

The current proposed framework design is tailored to fit the Framework for Security Level Evaluation (F4SLE) and CHESS project needs. Meaning, design decisions were affected by the framework, available project resources and the set timeframe. Therefore all possible design solutions were not analysed in Chapter 4.

Currently, the limited framework implementation was tested by a trusted pilot group. Therefore, some security measures were not implemented for the user interface, back end, and server. For example, security measures against automatisation attacks are missing, and the back end does not validate request content. The organisation representative data and aggregated results are stored in the server as plaintext, and no backup of the files exists. In order to use the implemented framework with a wider group, all measures identified in the threat analysis should be implemented.

6.2 Future work

The proposed framework design was validated using a pilot group (described in Chapter 5.7). The validation focused only on the self-assessment tool and assessing its usability. No validation regarding the administrative functionality was conducted by a sample of policymakers and framework managers. Therefore feedback regarding the server-based approach usability was not received.

Currently, the developed framework administrative functionality is solved with a server-based approach (described in Chapter 4.3). The framework manager has to manually create several files to establish a new version of F4SLE, and all calculations are solved with a predefine Jupyter Notebook file. For future work, the development and validation of an administrative user interface could be tackled.

The administrative interface would make managing questionnaire versions, benchmarks, and participant data more comfortable. In addition, statistics and graphs could be visualized in the interface and directly shared with relevant parties.

The implemented Measurement Application for Self-assessing Security is configured to allow responding to questions with a predefined answer scale. In addition, collected metadata fields with options are hardcoded in the UI. The further development of the MASS could focus on making the application dynamic. For example, the questionnaire file could define answer types and values. The questionnaire file could be used to construct the collected metadata. Implementing a more dynamic user interface allows MASS to be used in various ways: e.g., conducting threat audits and validating newly developed risk assessment frameworks.

Due to time restrictions, requirements (**R11, R30, R31, R33**) and threat analysis actions (Table 16) were not implemented in the limited proof of concept. Before utilizing the existing framework implementation, all requirements should be implemented. A penetration test could be utilised to verify the requirement implementation state and identify possible vulnerabilities.

References

[1] Riigi Teataja. *Eesti infoturbestandard*. https://www.riigiteataja.ee/akt/121122022034, (Accessed 27.03.2023). 2022.

[2] Information System Authority. *Tutvustus*. https://eits.ria.ee/et/avalehe-menueue/tutvustus/, (Accessed 27.03.2023). 2022.

[3] Information System Authority. *Kübertruvalisuse aastaraamat 2023*. https://www.ria.ee/media/2653/download, (Accessed 27.03.2023). 2023.

[4] Mari Seeba et al. *Security level evaluation with F4SLE*. 2023. DOI: https://doi.org/10.1145/3600160.3605045.

[5] Mari Seeba, Sten Mäses, and Raimundas Matulevičius. *Method for Evaluating Information Security Level in Organisations*. Ed. by Renata Guizzardi, Jolita Ralyté, and Xavier Franch. Cham, 2022. DOI: https://doi.org/10.1007/978-3-031-05760-1_39.

[6] Riigi Teataja. *Public Information Act*. https://www.riigiteataja.ee/en/eli/ee/502012023005/consolide/current, (Accessed 27.03.2023). 2023.

[7] Alan R. Hevner et al. "Design Science in Information Systems Research". In: *MIS Quarterly* 28.1 (2004), pp. 75–105. ISSN: 02767783. URL: http://www.jstor.org/stable/25148625 (visited on 06/27/2023).

[8] Information System Authority. *Eesti infoturbestandard*. https://eits.ria.ee/, (Accessed 03.08.2023). 2023.

[9] Information System Authority. *Standardist*. https://eits.ria.ee/et/avalehe-menueue/tutvustus/standardist/ (Accessed 27.06.2023). 2022.

[10] Information System Authority. *E-ITS v2022 (EN)*. https://eits.ria.ee/et/avalehe-menueue/eits-v2022-en/, (Accessed 04.05.2023). 2023.

[11] International Organization for Standardization. *ISO/IEC 27001:2022 Information security management systems. Requirements*. 2022.

[12] Mari Seeba. *Eesti Infoturbestandardi (E-ITS) põhine turbe hindamise instrument / Estonian Information security Standard (E-ITS) based security level evaluation instrument*. 2021. DOI: https://doi.org/10.23673/re-298.

[13] Mari Seeba. *Framework for Security Level Evaluation (F4SLE) E-ITS based ver 2021-1*. 2022. DOI: https://doi.org/10.23673/re-372.

[14] Mari Seeba et al. “Create your own MUSE: A method for updating security level evaluation instruments”. In: *Computer Standards Interfaces* 87 (2024), p. 103776. ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2023.103776.

[15] Cybernetica AS. *Andmekaitse ja infoturbe leksikon*. https://akit.cyber.ee/term/634, (Accessed 29.07.2023). 2023.

[16] K. Tuma, G. Calikli, and R. Scandariato. “Threat analysis of software systems: A systematic literature review”. In: *Journal of Systems and Software* 144 (2018), pp. 275–294. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2018.06.073. URL: https://www.sciencedirect.com/science/article/pii/S0164121218301304.

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

[18] Tarmo Oja. *X-road trsut model and technology threat analysis*. https://cyber.ee/uploads/tarmo_oja_msc_42196bb7e7.pdf, (Accessed 10.07.2023). 2020.

[19] Michael Muckin and Scott C Fitch. “A threat-driven approach to cyber security”. In: *Lockheed Martin Corporation* (2014). https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Threat-Driven-Approach.pdf, (Accessed 03.07.2023).

[20] Microsoft. *The STRIDE Threat Model*. https://learn.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20), (Accessed 04.07.2023). 2009.

[21] European Union Agency for Cybersecurity. *Cybersecurity Maturity Assessment for Small and Medium Enterprises*. https://www.enisa.europa.eu/cybersecurity-maturity-assessment-for-small-and-medium-enterprises, (Accessed 28.06.2023). 2023.

[22] Cybsis. *Eesti infoturbestandardi (E-ITS) rakendamise tööriist*. https://eits.raulwalter.com/, (Accessed 29.06.2023). 2023.

[23] Information System Authority. *Auditeerimisjuhend*. https://eits.ria.ee/et/versioon/2021/juhendid/auditeerimisjuhend/, (Accessed 03.07.2023). 2022.

[24] Information System Authority. *Riskihaldusjuhend*. https://eits.ria.ee/et/versioon/2022/juhendid/riskihaldusjuhend/, (Accessed 02.05.2023). 2022.

[25] Information System Authority. *APP.6: Tarkvara üldiselt*. https://eits.ria.ee/et/versioon/2022/etalonturbe-kataloog/app-rakendused/app6-tarkvara-ueldiselt/3-meetmed/#app6m10tarkvaraturvajuhend13, (Accessed 19.04.2023). 2022.

[26] Maria Pibilota Murumaa. *MASS*. https://mass.cloud.ut.ee/massui, (Accessed 09.08.2023). 2023.

[27] Raimundas Matulevičius. *Fundamentals of secure system modelling*. Springer, 2017. DOI: https://doi.org/10.1007/978-3-319-61717-6.

[28] Éric Dubois et al. "A Systematic Approach to Define the Domain of Information System Security Risk Management". In: *Intentional Perspectives on Information Systems Engineering*. Ed. by Selmin Nurcan et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 289–306.

[29] OWASP. *Top 10:2021*. https://owasp.org/Top10/, (Accessed 22.05.2023). 2021.

[30] Kaoru Ishikawa and Kaoru Ishikawa. *Guide to quality control*. Vol. 2. Asian Productivity Organization Tokyo, 1982.

[31] OWASP. *Application Security Verification Standard 4.0.3*. https://owasp.org/www-project-application-security-verification-standard/, (Accessed 09.08.2023). 2021.

Appendix 1 – Glossary

APT - Advanced Persistent Threat

CHESS - Cyber-security Excellence Hub in Estonia and South Moravia

CIA - Confidentiality, Integrity, and Availability

DB - Database

DoS - Denial of Service attack

E-ITS - Estonian Information Security Standard

ENISA - European Union Agency for Cybersecurity

F4SLE - Framework for Security Level Evaluation

FM - Framework Manager

MASS - Measurement Application for Self-assessing Security

MFA - Multi-factor authentication

MitM - Man-in-the-Middle attack

OrgR - Organisation Representative

OS - Operating System

PM - Policy Maker

PoC - Proof of Concept

RBAC - Role-Based Access Control

SA - Server Administrator

SAF - Self-Assessment Framework

SP - Software Provider

SSH - Secure Shell

UI - User Interface

VPN - Virtual Private Network

R1 -Content is available in multiple languages (e.g., Estonian and English)

R2 - The user can start measuring the security maturity without performing prior tasks (e.g., asset mapping or tool set up)

R3 - The measurement tool gives a detailed overview of results (e.g., for each module)

R4 - No account is needed to start the measuring process

R5 - The measurement tool provides a benchmark for result comparison

R6 - Collaboration between different roles is possible for the organisation's measuring process

R7 - The tool's content is updated regularly (e.g. each year)

R8 - The measurement tool includes the possibility to re-open the results

R9 - The measurement questionnaire is based on a common standard (e.g., ISO 27001, E-ITS)

R10 - Customised remediation plan is provided based on results

R11 - The security measuring is expected to take less than 2 hours to perform

R12 - The application must display a questionnaire containing ten 4-level module groups.

R13 - The user can evaluate each attribute on a 4-level scale, with the additional options of not answering or marking answer not applicable.

R14 - Detailed user input must not leave the user's local disk.

R15 - The application must display the results as follows: 4 radar diagrams displaying the results of each level and an overall radar diagram with benchmark overlay.

R16 -The application should be security sensitive.

R17 - The application should show the benchmarks only when the user has sent their results and additional data.

R18 - The user should be able to view the benchmarks without re-sending their results.

R19 - The UI should have a low entry barrier.

R20 - Submitting the F4SLE results should take as few steps as possible.

R21 - The user should be able to save answers to local disk.

R22 - The user should be able to continue answering an unfinished version.

R23 - The user should be able to quickly move between modules.

R25 - The application should visualize the answering progress.

R26 - The user should be able to leave text based feedback.

R27 - Results and relevant data should be sent to server with explicit consent.

R28 - The application should display the benchmark only when the user has submitted the results.

R29 - The component should allow the selection of benchmarks displayed in the user interface.

R30 - The policy maker should be able to access benchmark results, both visual and numeral format.

R31 - The policy maker should not be able to access detailed user data.

R32 - The framework manager should be able to access user submitted data.

R33 - The functionality must automatically calculate benchmarks based on selected types, for example, domain-based or nation-based.

R34 - The framework manager should be able to add, modify and delete F4SLE attributes.

R35 - The framework manager should be able to modify benchmark calculation and visualization rules.

R36 - Benchmark should be calculated based on 5 or more organisations' results belonging to the corresponding group.

Appendix 2 – License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Maria Pibilota Murumaa**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Designing a security sensitive self-assessment framework: Estonian use case,

   supervised by Mari Seeba and Tarmo Oja.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Maria Pibilota Murumaa
11/08/2023

Appendix 3 – MASS user experience form

Fig.1 shows the CryptPad[9] form for gathering input about the security maturity assessment application MASS user experience.

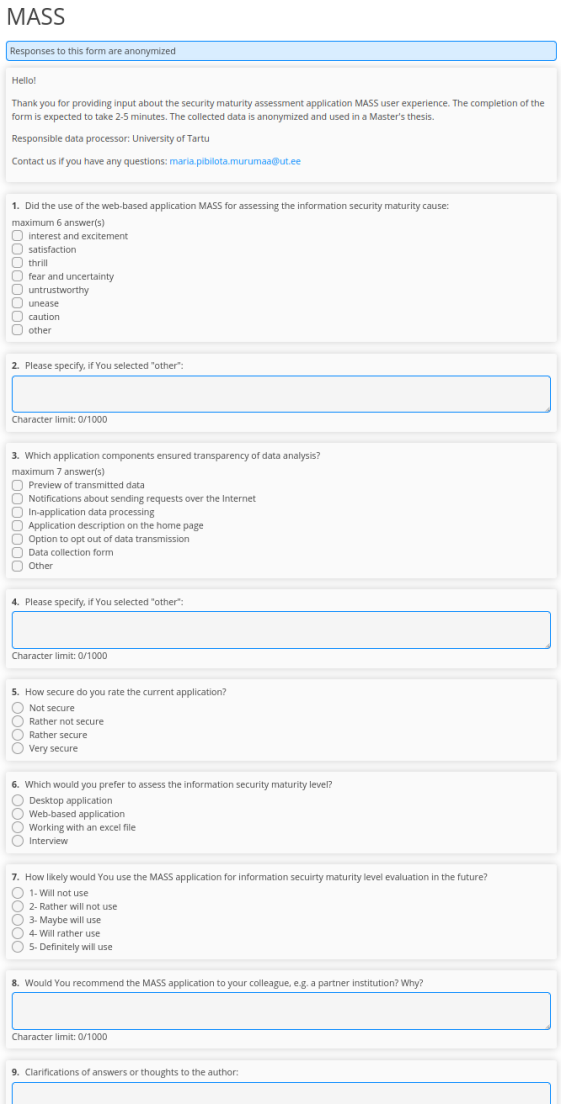

Figure 1. Screenshot of MASS user experience form

[9] CryptPad `https://cryptpad.fr/` (Accessed 09.08.2023)

Appendix 4 – Screenshots of MASS tool

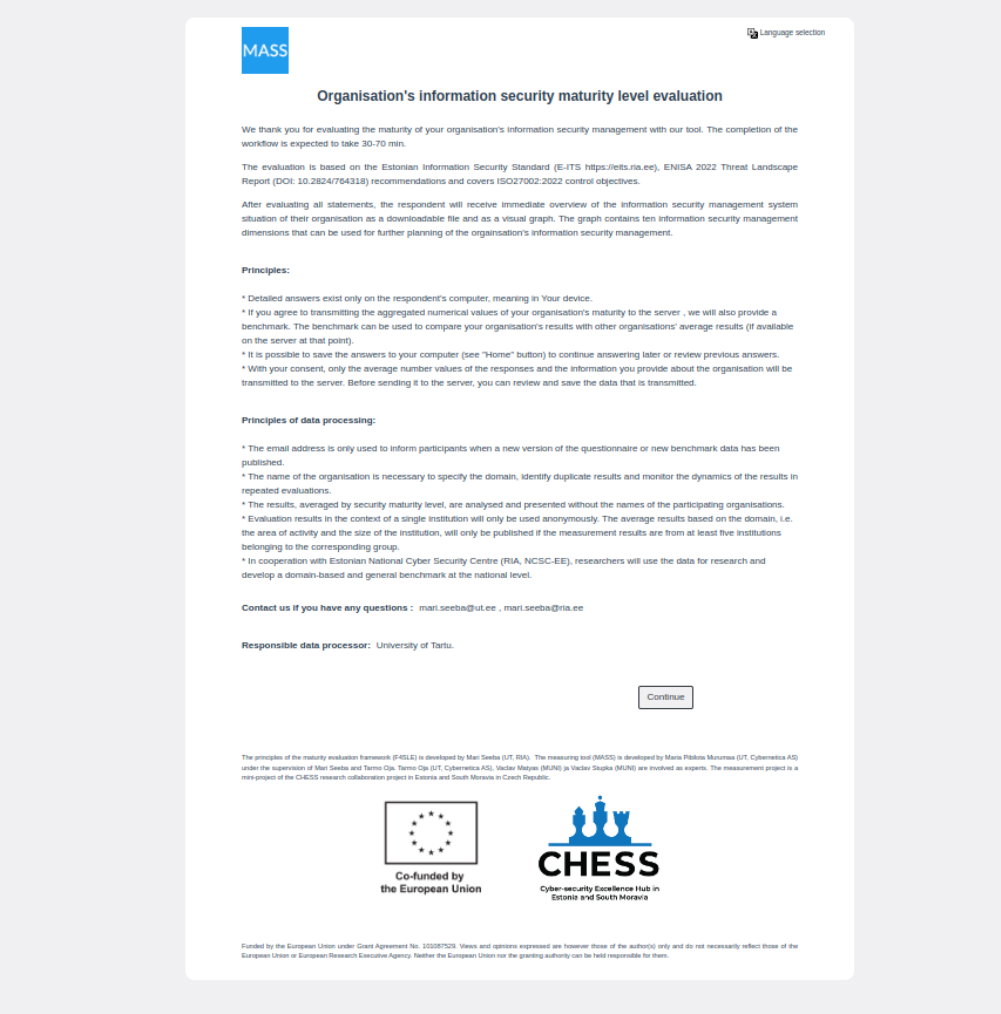

Figure 2. Screenshot of MASS landing page [26]

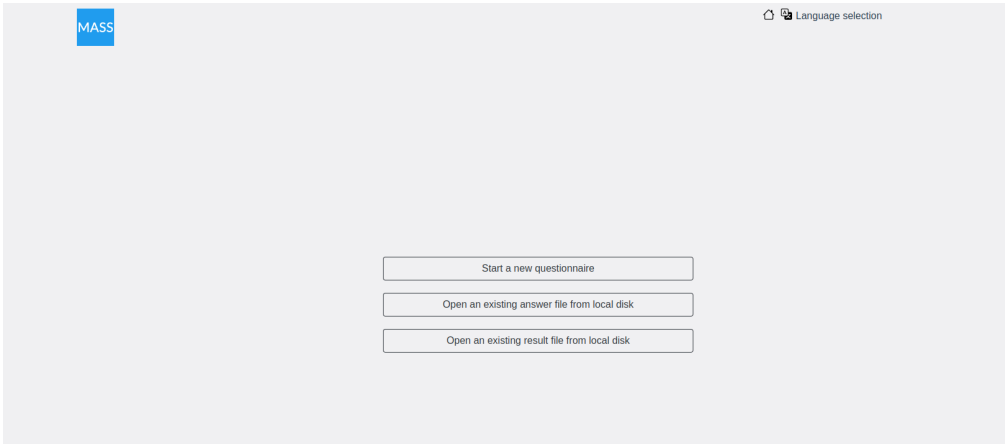

Figure 3. Screenshot of MASS options page [26]

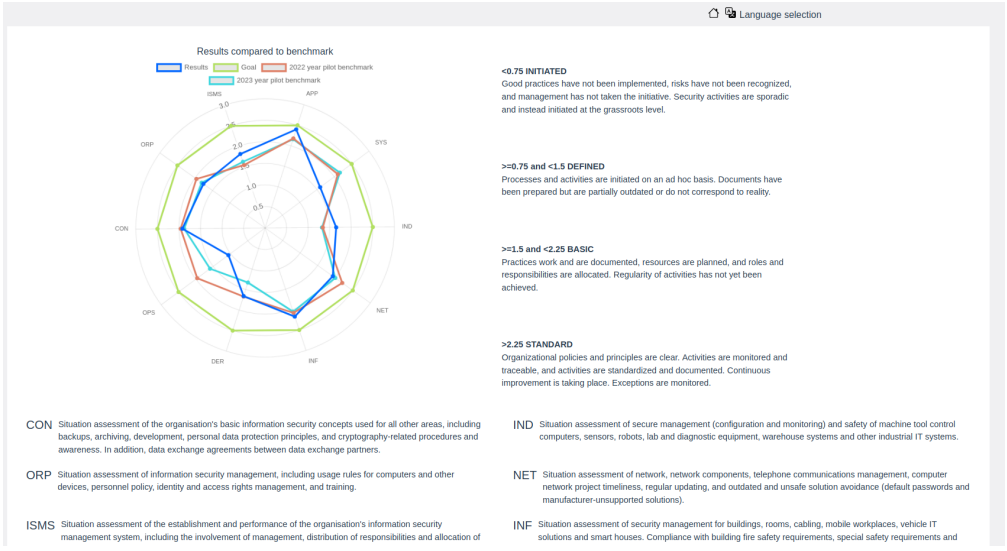

Figure 4. Screenshot of MASS result page first fragment [26]

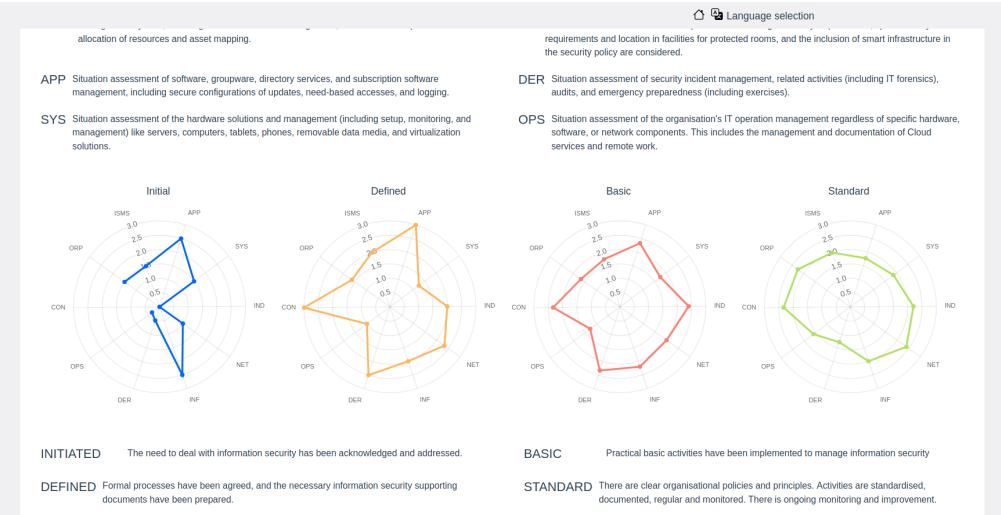

Figure 5. Screenshot of MASS result page second fragment [26]

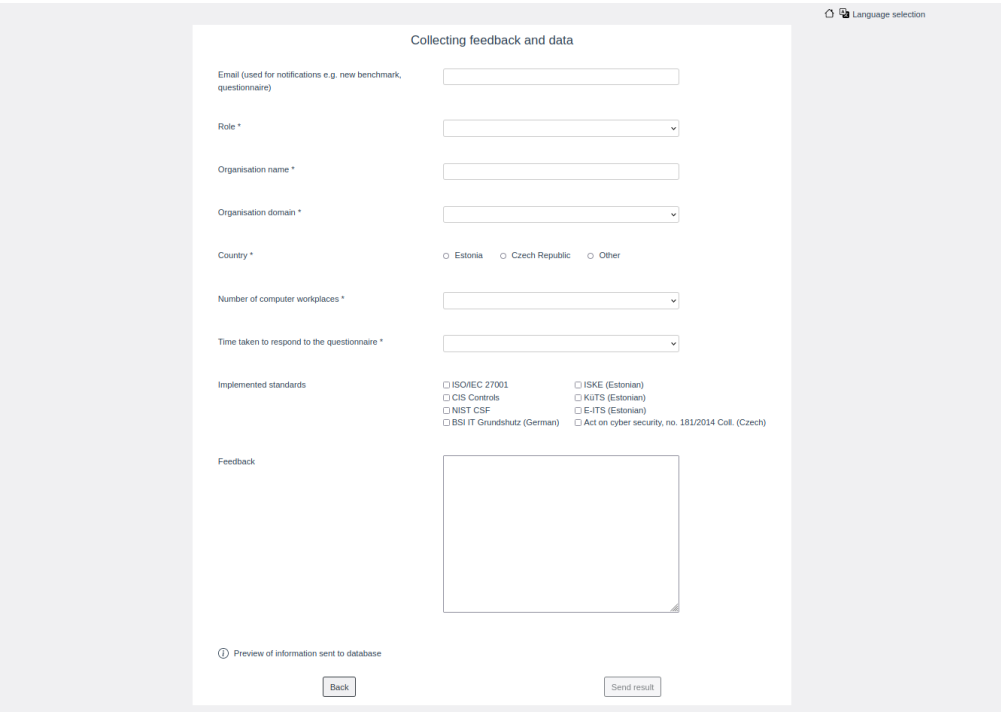

Figure 6. Screenshot of MASS data collection form [26]

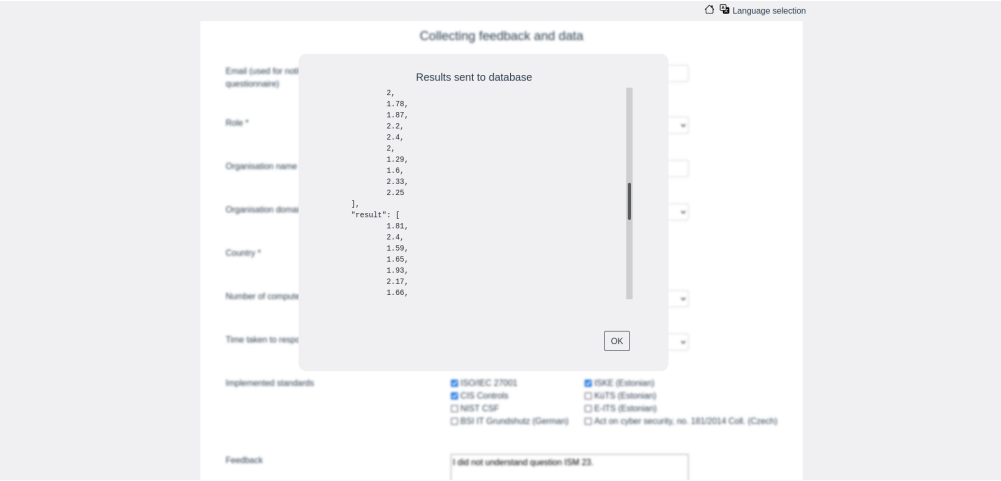

Figure 7. Screenshot of data preview [26]

Appendix 5 – Metadata

Table 1. Metadata collected with MASS.

Collected data List		
Data	Mandatory	Notes
Email	No	Used for notifications e.g. new benchmark, questionnaire), optional field.
Role	Yes	User can choose between 7 options: - Top management; - Auditor; - Information security manager /specialist; - IT manager, Network/system administrator; - Administrative assistant/lawyer/DPO; - Other.
Organisation name	Yes	Used for managing multiple submissions.
Organisation domain	Yes	User can choose between 8 options: - Healthcare; - Education; - Municipality; - ICT; - Private sector; - Non-profit; - Government office; - Other.
Country	Yes	User can choose between 3 options: - Estonia; - Czech Republic; - Other.

Number of computer workplaces	Yes	User can choose between 5 options: - 1...30; - 31...100; - 101...300; - 301...1000; - 1001...
Time taken to respond to the questionnaire	Yes	User can choose between 6 options: - Around 30 minutes; - Around 1 hour; - 2 hours; - 2-4 hours; - 4-8 hours; - More than 1 working day.
Implemented standards	No	User can choose up to 8 options: - ISO/IEC 27001; - NIST CSF; - CIS Controls; - ISKE (Estonian); - KÜTS (Estonian); - E-ITS (Estonian); - BSI IT Grundshutz (German); - Act on cyber security, no. 181/2014 Coll. (Czech).
Feedback	No	Text field for user comments and feedback regarding the questions and tool (**R26**).

Appendix 6 – Asset threat profiles

Metadata

The metadata asset consists of benchmark and answer address. Both locations are used for indicating the request endpoint addresses.

Table 2. Threat profile: Questionnaire metadata

	Description
Asset	**Questionnaire metadata**
Threat types	STI
Attack surface	Network Server Back end
Attack vectors	Development flaw: misconfiguration Manipulating server or back-end Modifying metadata in transit with MitM Gaining permissions: FM/SP/SA role
Threat actors	Malicious or uncaring FM/PM/SP/SA APT or script kiddie

Organisation representative data

Participant feedback and metadata (e.g. organisation domain, implemented policies) are all part of the organisation representative data asset.

Table 3. Threat profile: Organisation representative data

	Description
Asset	**Organisation representative data**
Threat types	STI
Attack surface	MASS Server Back end Network
Attack vectors	Development flaw: server or MASS leaks data, configuration flaw Human error Social engineering Modified participant answers with MitM Manipulating server, back-end or MASS Sending aggregated results as another OrgR Gaining permissions: FM/SA/SP role, source code Flooding the service with requests Injecting unsanitized data to MASS
Threat actors	Malicious or uncaring FM/PM/SP/SA/OrgR APT or script kiddie

Aggregated results

Participating organisation's 10x4 aggregated results of each F4SLE module.

Table 4. Threat profile: Aggregated results

	Description
Asset	**Organisation representative data**
Threat types	STI
Attack surface	MASS Server Back end Network
Attack vectors	Development flaw: server or MASS leaks data, configuration flaw Human error Social engineering Modified participant answers with MitM Manipulating server, back-end or MASS Sending aggregated results as another OrgR Gaining permissions: FM/SA role Flooding the service with requests Injecting unsanitized data to MASS
Threat actors	Malicious or uncaring FM/PM/SP/OrgR APT or script kiddie

Questionnaire

Term questionnaire is used to describe the set of different version of F4SLE files.

Table 5. Threat profile: Questionnaire

	Description
Asset	**Questionnaire**
Threat types	STRE
Attack surface	Administration activities Server Network
Attack vectors	Development flaw: misconfiguration Modifying data in transit with MitM Gaining permissions: FM/SA role Human error
Threat actors	Malicious or uncaring FM/SP/SA APT or script kiddie

Benchmark

Benchmark asset consists of the calculated benchmark based on aggregated results.

Table 6. Threat profile: Benchmark

	Description
Asset	**Benchmark**
Threat types	ST
Attack surface	Administration activities Server and configuration Network MASS Back end
Attack vectors	Development flaw: misconfiguration Modifying data in transit with MitM Gaining permissions: FM/SA role Injecting unsanitized data to MASS
Threat actors	Malicious or uncaring FM/SP/SA/OrgR APT or script kiddie

Token

The token is automatically generated and distributed to the organisation representative when valid results are sent to the server from MASS.

Table 7. Threat profile: Token

	Description
Asset	**Token**
Threat types	STI
Attack surface	Administration activities Server and configuration Network MASS Back end
Attack vectors	Development flaw: misconfiguration Modifying data in transit with MitM Gaining permissions: FM/SA role Injecting unsanitized data to MASS
Threat actors	Malicious or uncaring FM/SP/OrgR/SA APT or script kiddie

Environment

The server is used to host the web-application MASS and back-end functionality, store questionnaire and token data.

Table 8. Threat profile: Server and back end

	Description
Asset	**Environment (server and back end)**
Threat types	STRIDE
Attack surface	Server OS and services Configuration Network MASS GitLab code repository Back end
Attack vectors	Development flaw: misconfiguration, lack of mantainence or monitoring Gaining permissions: FM/SA role, code repository Vulnerable dependencies Flooding server resources Modifying source code build and upload to host server with MitM Unsanatized injections in MASS
Threat actors	Malicious or uncaring FM/OrgR/SP/SA/PM APT or script kiddie

Software and distribution

In order to gather data for F4SLE, the organisation representative, framework manager and policy maker rely on the software producer to develop and distribute MASS.

Table 9. Threat profile: MASS

	Description
Asset	**MASS**
Threat types	STRID
Attack surface	Source code and used dependencies Hosting environment Repository server and access information MASS
Attack vectors	Development flaw: misconfiguration, human error Gaining permissions: source code, server administrative account, repository, developer account Vulnerable outdated dependencies Modify repository or its content MitM on source code build and upload to host server Unhandled injection in MASS user interface Flooding server with requests to deny access to MASS
Threat actors	Malicious or uncaring SP/FM/SA APT or script kiddie Malicious dependency provider

Jupyter notebook

Jupyter notebook file is used for benchmark calculation, data analysis and visualisation.

Table 10. Threat profile: Jupyter notebook

	Description
Asset	**Jupyter notebook**
Threat types	STRIDE
Attack surface	Administration activities Server OS and services Configuration
Attack vectors	Development flaw: misconfiguration Gaining permissions: FM/SP role, server administrative account Human error Flooding the server with requests
Threat actors	Malicious or uncaring FM/PM/SP/SA APT or script kiddie
Controls	Logging Role based access controls with strong authentication mechanisms Privileged account protections Anti-automation controls Patching

Framework manager role

FM role is responsible for the creation and modification F4SLE files, participant data, Jupyter Notebook file, benchmark graph(s) and data.

Table 11. Threat profile: Framework manager role

	Description
Asset	**Framework manager role**
Threat types	STRE
Attack surface	Administration activities Server Configuration
Attack vectors	Development flaw: misconfiguration Gaining permissions: FM/SP/SA role, server administrative account Human error
Threat actors	Malicious or uncaring FM/PM/SP/S APT or script kiddie

Policy maker role

Individual with the PM role can view F4SLE files, Jupyter Notebook files, benchmark graph(s) and data. The PM role does not have any writing permissions.

Table 12. Threat profile: Policy maker role

	Description
Asset	**Policy maker role**
Threat types	STRE
Attack surface	Administration activities Server Configuration
Attack vectors	Development flaw: misconfiguration Gaining permissions: PM/SP/SA role, server administrative account Human error
Threat actors	Malicious or uncaring PM/SP/SA APT or script kiddie

Software provider role

Table 13. Threat profile: Software provider role

	Description
Asset	**Software provider role**
Threat types	STRE
Attack surface	Administration activities Server Configuration GitLab code repository
Attack vectors	Development flaw: misconfiguration Gaining permissions: code repository account Human error
Threat actors	Malicious or uncaring SP APT or script kiddie

GitLab code repository

The GitLab code repository is an external development platform used for storing code, deploying the web-application and the back end.

Table 14. Threat profile: GitLab code repository

	Description
Asset	**GitLab code repository**
Threat types	STRID
Attack surface	Network Software provider
Attack vectors	Human error
Threat actors	Malicious or uncaring SP APT or script kiddie

Detailed F4SLE answers

F4SLE answers are user provided assessment for each F4SLE attribute. The anwers are used by MASS to calculate aggregated results. This data is not sent to server nor stored in browser cache.

Table 15. Threat profile: Detailed F4SLE answers

	Description
Asset	**Detailed F4SLE answers**
Threat types	STI
Attack surface	Network Software provider MASS GitLab code repository
Attack vectors	Development flaw, human error Spoofing MASS Gaining permissions to the code repository account MitM on source code build and upload to host server Vulnerable outdated dependencies
Threat actors	Malicious or uncaring SP APT or script kiddie

Appendix 7 – Protocol flows

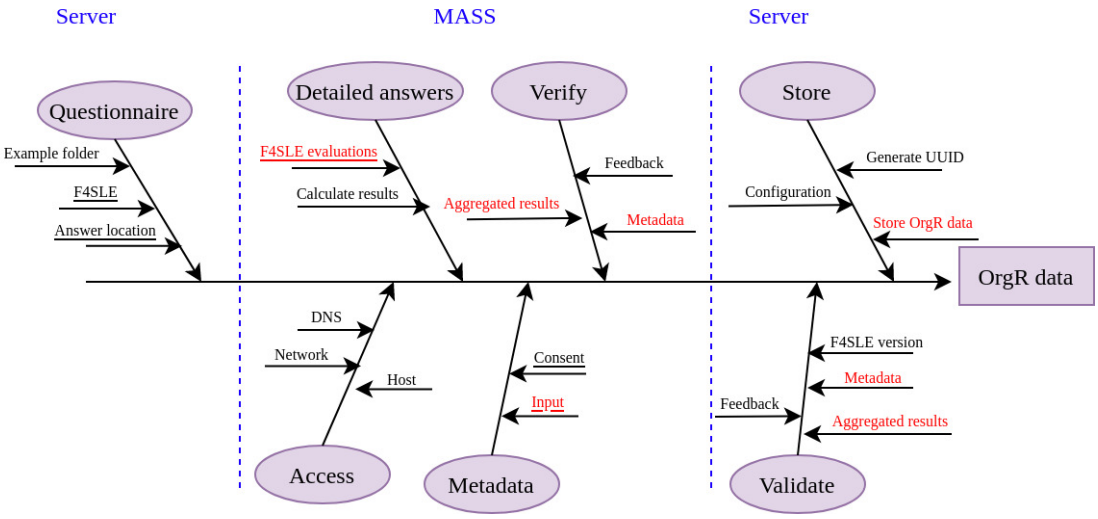

Figure 8. Protocol flow: Organisation representative data

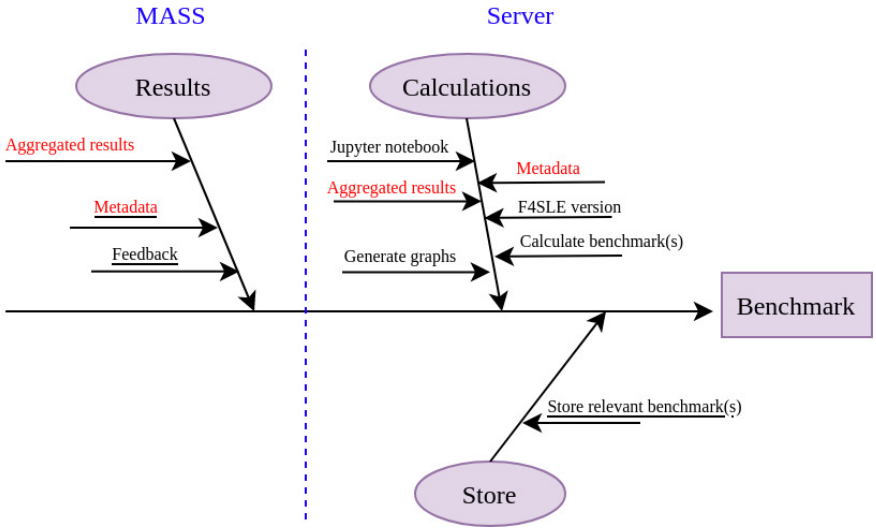

Figure 9. Protocol flow: Benchmark

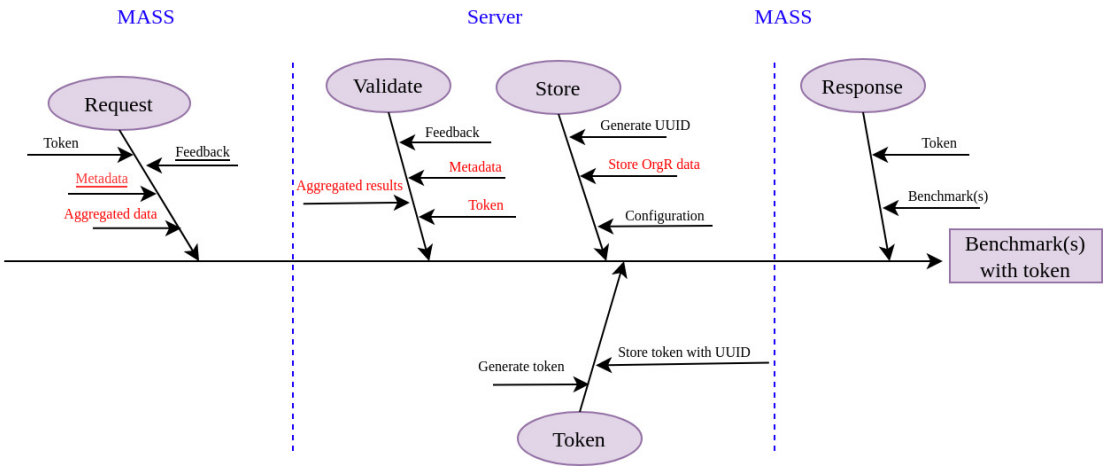

Figure 10. Protocol flow: Benchmark(s) and token

Appendix 8 – Threat analysis

Table 16. Threat analysis

ID	Target	STRIDE	Threat	Action
S01	MASS	S	An attacker spoofs dependencies used by MASS	**S01-A01:** All dependencies should come from trusted and mantained sources. **S01-A02:** All used dependencies (including their versions) should be listed. **S01-A02:** All used dependencies should be sandboxed.
S02	Aggregated results	S	Attacker pretends to be another OrgR	**S02-A01:** Duplicate answers or false answers can be manually evaluated by FM. **S02-A02:** Answers to F4SLE are non-personalized. **S02-A03:** All valid requests containing aggregated results will be stored on the server.
S03	MASS	S	An attacker spoofs MASS with different functionality	**S03-A01:** Single domain is used for all framework related pages. S03-A02: Monitor similar domain registrations.

S04	Code repository	S	An attacker spoofs code (e.g. new version of MASS or back end) upload to server (c1)	**S04-A01:** Communication takes place over authenticated encrypted channels. **S04-A02:** Server has firewall enabled with correct rules. **S04-A03:** IP whitelisting is enabled. S04-A04: Uploaded code is signed and the signature validated by the server. **S04-A05:** Application build and deployment processes should be automated and performed in a secure way.
S05	OrgR data	S	Attacker pretends to be another OrgR	**S05-A01:** When the back end receives an organisation's duplicate results, existing data will not be overwritten. S05-A02: Residual risk risk will be accepted.
T01	c2: Connection between MASS and back end	T	An attacker tampers the data exchanged between MASS and back end	**T01-A01:** Communication takes place over authenticated encrypted channels. T01-A02: Requests are validated on the server side to be the exact structure and type. **T01-A03:** Requests are sanitized on the server side. **T01-A04:** MASS contains detailed user answer file download and upload functionality to re-submit data.

T02	c1: connection between code repository and server	T	An attacker tampers the code (MASS, back end) when uploaded to server	**T02-A01:** Communication takes place over authenticated encrypted channels. **T02-A02:** Server has firewall enabled with correct rules. **T02-A03:** IP whitelisting is enabled. T02-A04: Uploaded code is signed and the signature validated by the server. **T02-A05:** Application build and deployment processes should be automated and performed in a secure way
T03	Code repository	T	Attacker (e.g. malicious SP) tampers the MASS code.	T03-A01: A new version of MASS code will not be released or deployed before conducting a threat analysis or code review. **T03-A02:** Multi-factor authentication (MFA) should be enabled to access the code repository. **T03-A03:** Threat is transferred to the 3rd party (GitLab)
T04	Code repository	T	Attacker (e.g. malicious SP) tampers the back end code.	T04-A01: A new version of back end code will not be released or deployed before conducting a threat analysis or code review. **T04-A02:** Multi-factor authentication (MFA) should be enabled to access the code repository. **T04-A03:** Threat is transferred to the 3rd party (GitLab

T05	MASS	T	An attacker tampers MASS to behave differently than intended by injecting malicious payload	**T05-A01:** All MASS input is validated to be the expected structure and type. T05-A02: All MASS user input is sanitized. T05-A03: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. **T05-A04:** MASS application should run in the server with lest required privileges. T05-A05: All exceptions should be handled with a general error message.
T06	Back end	T	An attacker tampers back end to behave differently than intended by injecting malicious payload	T06-A01: All received requests should be validated to be the expected structure and type. **T06-A02:** All received requests should be sanitized. T06-A03: Secure by design, secure by default and secure by deployment approach should be implemented throughout the framework. T06-A04: The back end should run with least required privileges in the server. T06-A05: All exceptions should be handled with a general error message.

T07	s1: Data stored in server	T	An attacker tampers with data stored in server by injecting a malicious payload	T07-A01: All received requests should be validated to be the expected structure and type. **T07-A02:** All received requests should be sanitized. T07-A03: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. T07-A04: The back end should run with least required privileges in the server. T07-A05: All exceptions should be handled with a general error message. T07-A06: OrgR data and aggregated answers should be encrypted by rest. T07-A07: OrgR data and aggregated answers should be moved to a separate location. **T07-A08:** RBAC should be defined. T07-A09: Least privileges for each role should be implemented.

T08	s1: Data stored in server	T	A malicious server user tampers with data stored in server	**T08-A01:** RBAC is implemented in the server. T08-A02: Least privileges should be implemented for each role. T08-A03: Role activity should be securely logged. T08-A04: OrgR data and aggregated answers should be encrypted by rest. T08-A05: OrgR data and aggregated answers should be moved to a separate location.
R01	Back end	R	The back end denies activity or information	**R01-A01:** Users are able to download their F4SLE answers and upload the results at a later time. R01-A02: Back end activity is securely logged to help find root of possible issues.
R02	Server	R	An attacker takes an action in the server and denies it	R02-A01: Server activity is securely logged to find root of possible issues.
R03	MASS	R	An attacker takes an action in MASS and denies it	**R03-A01:** Threat accepted.
R04	Code repository	R	An attacker takes an action in the code repository and denies it	**R04-A01:** Threat transferred to 3rd party (GitLab).

I01	MASS	I	MASS leaks data (development flaw)	**I01-A01:** User answers are not logged or stored in browser cache. **I01-A02:** Minimal input form user is collected and sent to server. I01-A03: All F4SLE modules must include at least two attributes to avoid revealing an attribute's exact evaluation. I01-A04: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. I01-A05: MASS is OWASP ASVS 4.03 level 2 compliant [31]
I02	Detailed F4SLE answers	I	Attacker leaks detailed FASLE answers	**I02-A01:** User answers are not logged or stored in browser cache. **I02-A02:** Minimal input form user is collected and sent to server. I02-A03: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. **I02-A04:** OrgR can check the data that is collected and sent to the server. **I02-A05:** OrgR can refuse from data collection.

I03	Detailed F4SLE answers	I	Detailed FASLE answers stored on the OrgR local hard drive leak to attacker	**I03-A01:** Downloadable answer files should not contain any indication to the organisation. **I03-A02:** The OrgR is responsible for the security and secrecy of the detailed F4SLE answers stored on the local hard drive.
I04	Aggregated results	I	Aggregated results stored on the OrgR local hard drive leak to attacker	I04-A01: Downloadable aggregated results file should not contain any indication to the organisation. I04-A02: The OrgR is responsible for the security and secrecy of the aggregated results stored on the local hard drive.
I05	c2: Connection between MASS and back end	I	Attacker listens to communication between MASS back end	**I05-A01:** Communication takes place over authenticated encrypted channels. I05-A02: Network is monitored for unusual traffic.

I06	s1: Data stored in server	I	Server leaks data (development flaw)	I06-A01: OrgR data and aggregated results are moved from the server to a separate location. **I06-A02:** Minimal input form the OrgR is collected. I06-A03: All F4SLE modules must include at least two attributes to avoid revealing an attribute's exact evaluation. I06-A03: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. I06-A04: Organisation representative and aggregated results data is encrypted at rest. **I06-A05:** Threat of disclosure of questionnaire metadata, questionnaire, benchmark data is accepted. I06-A06: Jupyter notebook kernel should be restarted after use.

I07	s1: Data stored in server	I	Malicious adversary (e.g. rouge FM, PM, SD, SA) leaks data stored in server	I07-A01: OrgR data and aggregated results are moved from the server to a separate location. **I07-A02:** Minimal input form the OrgR is collected. I07-A03: All F4SLE modules must include at least two attributes to avoid revealing an attribute's exact evaluation. I07-A03: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. I07-A04: OrgR and aggregated results data is encrypted at rest. **I07-A05:** Role based access controls (RBAC) are defined for server roles. **I07-A06:** RBAC can be accessed and changed only by the system administrator. **I07-A07:** Access to OrgR data and aggregated results is limited to FM role and SA role.
I08	Back end	I	Malicious adversary (e.g. rouge FM, PM, SD, SA) leaks back end code	I08-A01: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. **I08-A02:** Back end code should not include any sensitive information.

I09	GitLab code repository	I	Attacker leaks MASS code	I09-A01: Secure by design, secure by default and secure by development approach should be implemented throughout the framework. **I09-A02:** MASS code should not include any sensitive information.
I10	MASS	I	Attacker captures MASS input (e.g. screen recording, key logging)	**I10-A01:** Threat is accepted.
I11	c1: connection between code repository and server	I	Attacker listens to communication between the code repository and server.	**I11-A01:** Communication takes place over authenticated enrcyrpted channels. I11-A02: Network is monitored for unusual traffic.
I12	Code repository	I	Code repository leaks data (MASS code, back end code)	**I12-A01:** Threat action is transferred to the 3rd party (GitLab).
I13	Token	I	OrgR token is leakef	**I13-A01:** Tokens are linked to file names (randomly generated identifiers) and not directly to a organisation. **I13-A02:** Using a valid token to request benchmarks does not reveal any kind of information about the organisation.

D01	c2: Connection between MASS and back end	D	The attacker overloads the communication channel between MASS and back end	D01-A01: Back end activity is securely logged to find root of possible issues. D01-A02: Anti-automation controls are implemented for the server (e.g. rate limiting, IP blacklisting). D01-A03: Network traffic is monitored. D01-A04: Server uses a load balancer. **D01-A05:** Firewall is enabled in the server with correct rules.
D02	Back end	D	The attacker overloads the back end's communication channels	D01-A01: Back end activity is securely logged to find root of possible issues. D02-A02: Anti-automation controls are implemented for the server (e.g. rate limiting, IP blacklisting). D02-A03: Network traffic is monitored. D02-A04: Server uses a load balancer. **D02-A05:** Firewall is enabled in the server with correct rules.
D03	Back end	D	Back end blocks access and does not receive nor respond to requests	D03-A01: Back end activity is securely logged to find root of possible issues. **D03-A02:** MASS contains detailed user answer file download and upload functionality to re-submit data.

D04	MASS	D	Attacker uploads large size/amount of files to MASS	**D04-A01:** All user uploaded files are handled on the clients side and not loaded to the server **D04-A02:** MASS applies file upload restrictions (e.g. size, type)
D05	s1: Data stored in server	D	Attacker deletes all data stored in the server (e.g. questionnaire file)	**D05-A01:** RBAC should be defined. D05-A02: Least privileges for each role should be implemented. D05-A03: Data should be backed up and stored securely.
E01	Server	E	Attacker bypasses access controls	E01-A01: Require MFA for server connections. E01-A02: Harden the system to prevent root privilege escalation. E01-A03: Monitor abnormal process calls, new account creations **E01-A04:** Limit usage of system administrator account for day to day operations E01-A05: Ensure proper security configuration for server (e.g. restrict access to scripting components, network access, failed login attempts) **E01-A06:** Require IP whitelisting. **E01-A07:** Require the usage of a virtual private network (VPN).

E02	Code repository	E	Attacker bypasses access controls	E02-A01: Require MFA for server connections. **E02-A02:** Require IP whitelisting. **E02-A03:** Require the usage of a virtual private network (VPN). **E02-A04:** Threat is transferred to 3rd party (GitLab).