UNIVERSITY OF TARTU
Institute of Computer Science
Software Engineering Curriculum

Silver Samarütel

Revision of Security Risk-oriented Patterns for Distributed Systems

Master's Thesis (30 ECTS)

Supervisor: Raimundas Matulevičius

Tartu 2016

Revision of Security Risk-oriented Patterns for Distributed Systems

Abstract:

Security risk management is an important part of software development. Given that majority of modern organizations rely heavily on information systems, security plays a big part in ensuring smooth operation of business processes. Many people rely on e-services offered by banks and medical establishments. Inadequate security measures in information systems could have unwanted effects on an organization's reputation and on people's lives. Security concerns usually need to be addressed throughout the development and lifetime of a software system. Literature reports however, that security is often considered during implementation and maintenance stages of software development. Since security risk mitigation usually results with changes to an IS's specification, security analysis is best done at an early phase of the development process. This allows an early exclusion of inadequate system designs. Additionally, it helps prevent the need for fundamental and expensive design changes later in the development process. In this thesis, we target the secure system development problem by suggesting application of security risk-oriented patterns. These patterns help find security risk occurrences in business processes and present mitigations for these risks. They provide business analysts with means to elicit and introduce security requirements to business processes. At the same time, they reduce the efforts needed for risk analysis. We confront the security risk-oriented patterns against threat patterns for distributed systems. This allows us to refine the collection of existing patterns and introduce additional patterns to mitigate security risks in processes of distributed systems. The applicability of these security risk-oriented patterns is validated on business processes from aviation turnaround system. The validation results show that the security risk-oriented patterns can be used to mitigate security risks in distributed systems.

Keywords:

Security risk management, security patterns, security requirements engineering

CERCS:

T120 Systems engineering, computer technology

Turvariskidele orienteeritud mustrite ümbertöötamine hajussüsteemidele

Lühikokkuvõte:

Turvariskide haldamine on oluline osa tarkvara arendusest. Arvestades, et enamik tänapäeva ettevõtetest sõltuvad suuresti infosüsteemidest, on turvalisusel oluline roll sujuvalt toimivate äriprotsesside tagamisel. Paljud inimesed kasutavad e-teenuseid, mida pakuvad näiteks pangad ja haigekassa. Ebapiisavatel turvameetmetel infosüsteemides võivad olla soovimatud tagajärjed nii ettevõtte mainele kui ka inimeste eludele.

Tarkvara turvalisusega tuleb tavaliselt tegeleda kogu tarkvara arendusperioodi ja tarkvara eluea jooksul. Uuringute andmetel tegeletakse tarkvara turvaküsimustega alles tarkvara arenduse ja hooldus etappidel. Kuna turvariskide vähendamine kaasneb tavaliselt muudatustena informatsioonisüsteemi spetsifikatsioonis, on turvaanalüüsi mõistlikum teha tarkvara väljatöötamise algusjärgus. See võimaldab varakult välistada ebasobivad lahendused. Lisaks aitab see vältida hilisemaid kulukaid muudatusi tarkvara arhitektuuris.

Käesolevas töös käsitleme turvalise tarkvara arendamise probleemi, pakkudes lahendusena välja turvariskidele orienteeritud mustreid. Need mustrid aitavad leida turvariske äriprotses-

sides ja pakuvad välja turvariske vähendavaid lahendusi. Turvamustrid pakuvad analüütikutele vahendit turvanõuete koostamiseks äriprotsessidele. Samuti vähendavad mustrid riskianalüüsiks vajalikku töömahtu. Oma töös joondame me turvariskidele orienteeritud mustrid vastu hajussüsteemide turvaohtude mustreid. See võimaldab meil täiustada olemasolevaid turvariski mustreid ja võtta kasutusele täiendavaid mustreid turvariskide vähendamiseks hajussüsteemides.

Turvariskidele orienteeritud mustrite kasutatavust on kontrollitud lennunduse äriprotsessides. Tulemused näitavad, et turvariskidele orienteeritud mustreid saab kasutada turvariskide vähendamiseks hajussüsteemides.

Võtmesõnad:

Tarkvara turvariskide haldamine, turvamustrid, turvanõuete koostamine

CERCS:

T120 Süsteemitehnoloogia, arvutitehnoloogia

Table of Contents

1 Introduction

"Security stands for the ability to protect information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction" (Andress, 2011). It is one of the most important software qualities. Given that majority of modern organizations rely heavily on information systems, security plays a big part in ensuring smooth operation of business processes. Many people rely on e-services offered by banks and medical establishments. For instance, in Estonia almost all medical prescriptions are handled through the e-Prescription system. Inadequate security measures in information systems could have unwanted effects on an organization's reputation and on people's lives.

Security concerns usually need to be addressed throughout the development and lifetime of a software system. Literature reports however, that security is often considered during implementation and maintenance stages of software development (Jürjens, 2005). Since security risk mitigation usually results with changes to an IS's specification, security analysis is best done at an early phase of the development process. This allows an early exclusion of inadequate system designs. Additionally, it helps prevent the need for fundamental and expensive design changes later in the development process.

"A pattern is a solution to a problem that arises within a specific context" (Buschmann *et al.*, 2007). Patterns first gained recognition in the software development community after the Gang-of-Four issued a book on design patterns (Gamma *et al.*, 1994). In this thesis we target the secure system development problem by suggesting application of the security risk-oriented patterns (Ahmed & Matulevičius, 2013). These patterns help find security risk occurrences in business processes and present mitigations for these security risks. They provide business analysts with means to elicit and introduce security requirements to business processes. More specifically we consider how security risk-oriented patterns could be used in distributed systems, such as an aviation turnaround system (Nõukas, 2015). We confront the security risk-oriented patterns against threat patterns for distributed systems (Uzunov & Fernandez, 2013). This allows us to refine the collection of existing patterns and introduce additional patterns to mitigate security risks in the processes of distributed systems.

The main research question of the thesis is the following:

RQ: How can the library of security risk-oriented patterns be expanded to support development of secure distributed systems?

This question is refined into three sub-questions:

RQ1: Which security threats do the security risk-oriented patterns correspond to?

To answer this question we consult the security threats library by Uzunov & Fernandez (2013) and align the security risk-oriented patterns to the security threats.

RQ2: Which additional security risks and countermeasures should be considered to be able to apply security risk-oriented patterns for securing business processes in distributed systems?

Threat patterns that do not align with the security risk-oriented patterns are potential candidates for new security risk-oriented patterns. We examine each unaligned threat pattern to determine if it could pose a risk on a business process level.

RQ3: What is the applicability of the security risk-oriented patterns in distributed systems?

To answer this question, we apply the security risk-oriented patterns to business processes from the aviation industry. After this we submit the resulting business processes enhanced with security requirements for expert review.

The thesis is structured as follows. Chapter 2 presents the background of security requirements engineering and gives an overview of several commonly used security engineering methodologies. Additionally, the chapter covers and compares common modelling languages used for security risk management. Chapter 3 provides the background for security risk-oriented patterns. The current state of the security risk-oriented patterns library is examined in more detail and proposals are given on how it could be improved. Chapter 4 is devoted to the contribution of the thesis. In this chapter, changes to the security risk-oriented patterns library are proposed. Chapter 5 covers the validation of the improvements proposed in chapter 4. In the concluding chapter 6 suggestions for future work and closing remarks are provided.

2 Security Requirements Engineering

In this chapter we define security engineering and security risk management activities. We introduce three prevalent security engineering approaches and cover some commonly used modelling languages for security risk management. We also examine the ISSRM domain model and explain how it assists us in addressing security risks.

2.1 Security Engineering

"Security engineering is an engineering discipline within systems engineering concerned with lowering the risk of intentional unauthorized harm to valuable assets to a level that is acceptable to the system's stakeholders by preventing, detecting and reacting to malicious harm, misuse (i.e., attacks and incidents), threats and security risks" (Firesmith, 2007).

Software development organizations ordinarily use some development process for software construction. Unfortunately, these development processes provide little or no support for security engineering. There are three high-profile approaches however, for introducing security concerns into the software development lifecycle. These are the Seven Touchpoints for Software Security (McGraw, 2006), Microsoft's Security Development Lifecycle (Howard & Lipner, 2006) and OWASP CLASP (Secure Software, Inc., 2005). In the following subchapter we shall give a brief overview of all three methodologies.

2.1.1 Security Engineering Approaches

Seven Touchpoints for Software Security. Security engineering concerns artefacts from all stages of a software development process. Introducing security engineering into software development requires changes to the way software is built. However, making fundamental changes to an existing and proved development process is not something any software development organization would likely wish to do. The Seven Touchpoints methodology offers a solution to this problem by integrating security best practices to an existing development process.

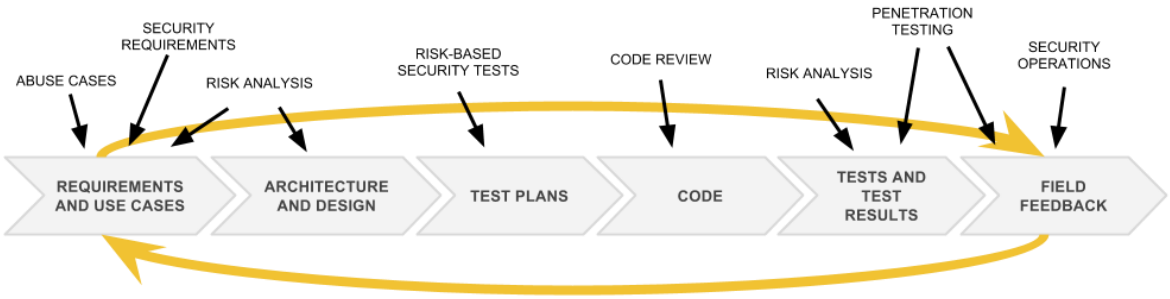


Figure 1. Seven Touchpoints: Security best practices applied to respective software artefacts, adapted from (McGraw, 2006)

Figure 1 depicts the six most common software development artefacts (i.e., requirements and use cases, architecture and design, test plans, code, tests and test results, and field feedback) together with respective touchpoints (i.e., security best practices). Although the figure resembles the waterfall process model, the Seven Touchpoints methodology is process independent, meaning it can be applied to any development process being used for software engineering (McGraw, 2006). We shall now examine each touchpoint in greater detail.

Code review is an essential best practice for creating secure software, especially since code is the one artefact that is present in all software products. The purpose of code review is to find implementation bugs. However, code review alone is not sufficient for creating secure

software, since security problems can be also caused by design flaws. Therefore, it is important to perform architectural risk analysis.

Figure 2. Security engineering approach, adapted from (Schumacher, 2003)

Architectural risk analysis is a risk management process for finding flaws in the software architecture and design. Figure 2 depicts the general course of architectural risk analysis. It begins with elicitation of all the assets that are to be protected. The next step is identification of threats and attacks that could fall upon the assets. Risk estimation is performed on all the identified threats and attacks, including estimations of potential losses if a specific attack were to be successful. Elimination of all risks is almost always unreasonable due to cost and time consumption. Therefore, risks are prioritized in the order of severity. Mitigations to risks are considered in the form of countermeasures. Since the countermeasures can potentially introduce new flaws, the process is repeated until a state of acceptable risk is achieved.

Penetration testing is a type of black box testing that is performed with the system in its real environment (McGraw, 2003). Passing a penetration test does not reveal much about the security of the software. However, failing a penetration test is a clear indication of severe defects in the software.

Risk-based security tests are performed using common attack patterns, risk analysis and abuse cases (McGraw, 2003). Knowledge of the system's architecture is needed.

Abuse cases describe how a software system behaves under an attack. They must be worked into the requirements artefact along with s*ecurity requirements*. Security requirements should cover both functional security and emergent characteristics.

Security operations can be applied to a system that has already been fielded. The fielded system is monitored for attacks. If a successful attack is made, software behaviour is studied and changed in order for similar attacks not to be successful in future.

Although not a separate touchpoint, external review is a practice recommended alongside the Seven Touchpoints. The reasoning behind it is that people are less likely to find faults in their own work and more likely to find them in others' work (McGraw, 2003).

Microsoft's Security Development Lifecycle. Much like the Seven Touchpoints, Microsoft's Security Development Lifecycle (SDL) can be integrated with an existing software development process in order to create secure software. The SDL process consists of seven phases, as depicted on Figure 3. These phases are as follows: training, requirements, design, implementation, verification, release and response. We shall examine each phase in more detail.

Microsoft believes education and awareness to be fundamental cornerstones for building secure software and while young software engineers coming from universities and colleges understand security features, they do not know how to build secure software (Howard & Lipner, 2006). At minimum, software developers should be familiar with best practices of

secure software design, threat modelling, secure coding, security testing and privacy. According to SDL, attending at least one security training per year should be mandatory. Trainings should constantly be updated to address newest threats, research and mitigations.

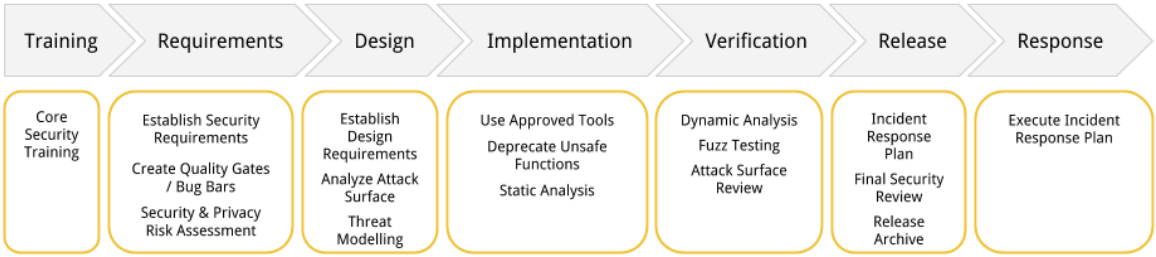


Figure 3. Security Development Lifecycle phases, adapted from (Howard & Lipner, 2006)

Requirements phase is concerned with establishment of security requirements and bug bars as well as assessment of security and privacy risks. Security and privacy risk assessments determine the parts of the software that require deep security review. For privacy risk assessment, the sensitivity of different data processed in the software is assessed to determine the Privacy Impact Rating. The Privacy Impact Rating represents the amount of work that is required to comply with the privacy requirements. In addition the project's bug bar is defined which sets the minimum level of security quality for the software to be built. The bug bar determines the types of bugs that shall be fixed and the types that shall remain as is.

Design phase consists of establishment of design requirements, analysation of attack surface and threat modelling. During this phase security and privacy design specifications are created and reviewed. Secure design best practices are applied, the most important being reduction of the attack surface. The attack surface is defined as the union of code, interfaces, services and protocols available to all users (Howard & Lipner, 2006). In addition, threat modelling is performed to identify potential security threats and to provide appropriate mitigations.

Implementation phase focuses on code quality. Static analysis of source code should be applied during this phase. Use of deprecated functions is strongly discouraged. Any external APIs that are deemed unsafe should be banned and replaced with safer alternatives. It is a good practice to define a list of approved tools to be used in the project. Using newest compilers and code scanning tools is strongly recommended wherever possible.

Verification phase ensures that the code meets the established requirements. This requires thorough testing of the software using fuzz testing, penetration testing, run-time verification, re-reviewing the threat models and re-evaluating the attack surface. The software must meet the security bar set during the requirements phase.

Release phase prepares the software for public release. During this phase an incident response plan is created. The plan prescribes the response process for when new threats emerge. SDL also foresees a Final Security Review (FSR) during this phase. FSR is an inspection of all the security activities performed prior to the release of the software. If the security team agrees that the FSR is successful, the product can be released.

Response phase executes the response plan conducted during the release phase whenever security vulnerability reports emerge. Having a response plan enables the development team to execute their response in a swift manner, without wasting time on making decisions or fearing about overlooking something. Most importantly, it helps protect the customers from the newest threats.

OWASP CLASP. CLASP (Comprehensive, Lightweight Application Security Process) is a role-based set of security best practices, which likewise the Seven Touchpoints and SDL can be integrated into an existing software development process. The core of CLASP are 24 security related activities which are assigned to specific roles in a software organization. We discuss the practices proposed by CLASP in more detail below.

Instituting awareness programs is a good way to educate people in a software organization about essential security concepts and techniques. Besides security training, establishing accountability for security issues is also necessary to raise security awareness. It also helps to promote a better outcome from subsequent security practices.

Performing application assessments helps in finding design, specification or implementation errors, thus reducing the number of security problems that make it into the software being built. Activities belonging to this practice include implementation and performing of security tests, threat modelling, source-level security review, and assessment of technology solutions' security posture.

Capturing security requirements should be an explicit part of development effort if security is of any importance at all in a software project. Activities within this practice include identification of attack surface, misuse cases, trust boundaries, and user roles' resource capabilities.

Implementing secure development practices should become a part of an organization's development culture. This CLASP best practice introduces activities that guide developers through applying security principles to design, implementing security functionality into the project's specification, as well as implementing and elaborating resource policies and security technologies.

Building vulnerability remediation procedures allows software engineers to effectively address security vulnerabilities that are discovered during the lifetime of the project. The remediation procedures should describe how vulnerabilities are assessed and prioritized, as well as how fixes are validated and delivered to the end-users.

Defining and monitoring metrics allows the development team to assess the security posture of the software being built. Metrics help determine the effectiveness of security measures and investments. They are also practical for pinpointing any inadequacies in the development process.

Publishing operational security guidelines enables the future maintainers of the system to make the most of the security measures implemented in the software. Providing good documentation on recommended operational security measures requires little effort, but can help secure the product more effectively.

Comparison of the Security Engineering Development Approaches

The three aforementioned processes address security engineering in different manners. In-depth analysis has been performed by (Buyens *et al*., 2007) where activities from each process were gathered and inserted into a single matrix. Table 1 depicts an adapted version of this activity matrix. The activities gathered from the development processes have been divided into nine regular development phases. Numbers in brackets following names of the development phases represent the total number of activities in the specific phase. While we cannot measure efficiency based on the activity matrix, we can see by comparing the numbers of activities from each process that CLASP and SDL are more heavyweight than Touchpoints. One of the main reasons for Touchpoints being notably less strict than the others is that it does not prescribe any activities for detailed design and support phases.

Although we might expect better security from the more rigorous processes, choosing the most optimal process really depends on the organization's needs and available resources. The Seven Touchpoints are more affordable and suitable for smaller organizations while larger organizations might consider applying Microsoft's SDL. Whichever process is chosen, all three processes have gone through validation and have proved to be effective for increasing software security (Buyens *et al*., 2007).

Table 1. Activity Matrix, adapted from (Buyens et al., 2007)

Development phase	SDL	CLASP	Touchpoints
Education and awareness (7)	5	3	1
Project inception (14)	6	8	4
Analysis and requirements (31)	1	17	14
Architectural design (24)	11	16	5
Detailed design (14)	6	8	0
Implementation (23)	8	18	1
Testing (23)	8	8	10
Release and deployment (6)	2	3	1
Support (13)	12	5	0
Total (155)	**59**	**86**	**36**

2.1.2 Security Risk Management

"Software risk management is a discipline whose objectives are to identify, address and eliminate software risk items before they become threats to successful software operation or major sources of expensive software rework" (Boehm, 1989).

Risk management is a broad term and it can address different types of risks. In this thesis the focus is essentially on security risks. Since resources are always limited, mitigating software security risks in a cost-effective manner can be a challenge. Software development organizations would benefit from eliminating or lowering risks in areas most essential to their business strategies in order to get most out of their investments. This can be done by comparing the cost of losses in case of a potential security breach to the cost of mitigating the enabling security risk.

Applying countermeasures to security risks reflects in an IS's specification. Postponing security risk management to latter parts of a software development process results in changes to the existing IS's specification, thus making the development more expensive and time-consuming. Therefore it is sensible to address security concerns during initial phases of the development process in order to discard inadequate design solutions as early as possible.

There are hundreds of risk management methodologies and multiple security modelling frameworks available for security risk management. Some prevalent examples are CORAS (Braber *et al*., 2007), EBIOS (DCSSI, 2004), CRAMM (Insight Consulting, 2003) and OCTAVE (Alberts & Dorofee, 2001). These methodologies provide general guidelines for

identifying vulnerable assets, determining security objectives, assessing risks as well as defining and implementing security requirements for treating security risks (Dubois *et al.*, 2010).

Risk management methods can at large be divided into two: qualitative and quantitative methods. Qualitative methods use classifications to rate a potential impact of a threat as high, medium or low. Quantitative methods on the other hand use two inputs: probability of an event occurring and the cost of potential losses (Behnia *et al.*, 2012). The product of these numbers is called Expected Annual Loss.

CORAS is a model-based qualitative security risk analysis method which comes with its own language and supporting tool for modelling risks. The method is based on academic research, empirical studies, thorough experience, as well as close interaction and cooperation with actors from industrial domain (Lund *et al.*, 2011).

EBIOS (Expression des Besoins et Identification des Objectifs de Sécurité) is a qualitative approach which was initially intended for governmental organizations. It is composed of five steps which are context study, expression of sensitivities, risk study, security objectives identification and security requirements determination (DCSSI, 2004).

CRAMM (CCTA Risk Analysis and Management Method) is a qualitative approach that consists of three stages: establishing security objectives, assessing risks and identification of countermeasures (Insight Consulting, 2003).

OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) is another qualitative approach for managing security risks. OCTAVE is focused on strategic and practice-related issues rather than technology. The primary target of OCTAVE are larger organizations with over 200 employees (Alberts & Dorofee, 2001).

Domain Model for Security Risk Management

The ISSRM (Information System Security Risk Management) domain model is based on analysis of security and risk management standards, methods and frameworks (Dubois *et al.*, 2010). It presents key concepts and their relationships used to define a security risk-based template (see Figure 4). These concepts are asset-related, risk-related and risk treatment-related.

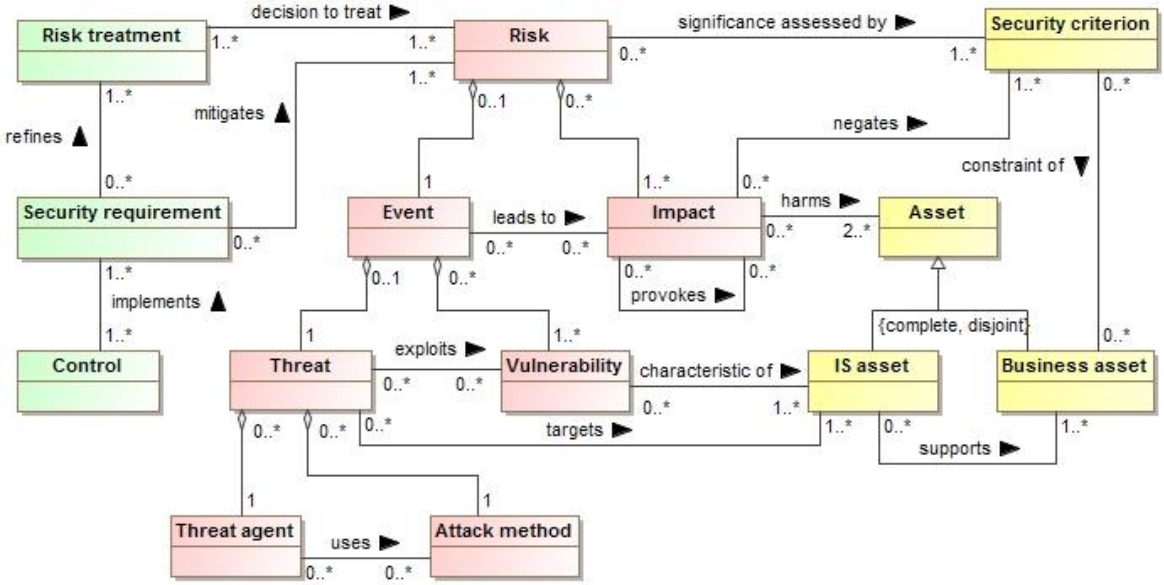


Figure 4. ISSRM Domain Model, adapted from (Dubois *et al.*, 2010)

Asset-related concepts represent assets of an organization and their security criteria. Assets are divided into business assets (e.g., information about processes, confidential data, and skills) and information system assets, i.e., the infrastructure supporting the business assets. Security criteria describe the security needs of the business assets, i.e., availability, confidentiality and integrity requirements.

Risk-related concepts describe the underlying relations between the concepts that form a risk. A risk is composed of an event that leads to one or more negative impacts. An impact negates one or more security criterion and as a result harms the assets. An event is a consequence of a threat exploiting a vulnerability in the system. A threat is a potential attack, carried out by a threat agent using an attack method.

Risk-treatment related concepts describe the relations between decisions, requirements and controls for treating risks. A risk treatment represents a decision how to treat a risk (e.g., by avoiding, reducing, transferring or retaining the risk). A security requirement is a refinement of a risk treatment that mitigates a risk. A control represents the means to implement a security requirement.

2.2 Modelling Languages for Security Risk Management

There are numerous modelling languages available for modelling security requirements. Some commonly used examples are Misuse Cases (Sindre & Opdahl, 2005), Mal-Activity Diagrams (Sindre, 2007), Secure Tropos (Mouratidis & Giorgini, 2007) and Business Process Model and Notation (Dijkman *et al*., 2007; Silver, 2009). In this chapter we give a brief overview of the languages and explain why we chose BPMN for presenting our contributions in this thesis.

2.2.1 Business Process Model and Notation

Business Process Model and Notation (BPMN) is a widely used modelling language for graphical representation of business processes. Versatility is one of the reasons behind its popularity in academia as well as industry. It is applicable for analytical, executable, as well as descriptive modelling (Matulevičius, 2012).

However, BPMN lacks possibilities for addressing security concerns when analysing business needs. Extensions for BPMN proposed by (Altuhhova *et al*., 2013) help conform BPMN for security risk management purposes. These extensions are based on alignment of BPMN to the ISSRM domain model. The resulting security risk-aware BPMN allows analysts to introduce security requirements into business processes modelled in BPMN. This is good, because business processes and their security concerns can be addressed at the same time.

Security risk-aware BPMN uses different colours for representing ISSRM concepts. Black is used for asset-related constructs, red is used for risk-related constructs and blue is used for risk-treatment related constructs. Table 2 highlights the main constructs that are used in the BPMN models in this thesis.

Table 2. Security Risk-aware BPMN Syntax, adapted from (Altuhhova *et al.*, 2013)

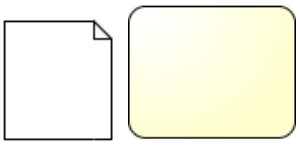
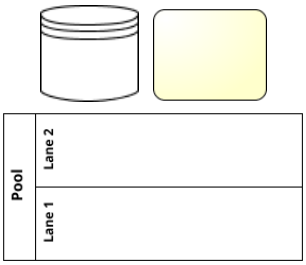
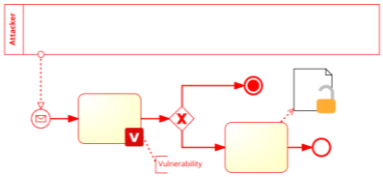
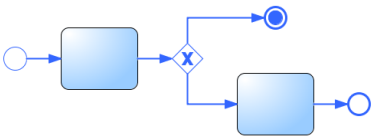
Asset-related Concepts		
ISSRM Concept	**BPMN Constructs**	**Syntax**
Business asset	Data object Task	
IS asset	Data store Task Pool and lanes	
Security objective	Property of a lock with a value from the CIA triad: c – confidentiality i – integrity a - availability	
Constraint	Lock describes the security constraints of a business asset (depicted as a data object or a task)	
Risk-related Concepts		
Impact	An unlock associated with a business asset describes an impact of an event	
Vulnerability	Annotation	
IS characteristic	Vulnerability is a characteristic of an IS asset (represented using a Task)	
Risk	Combination of flow objects (i.e., tasks, gateways, events), sequence flows, data flows, vulnerabilities, IS characteristics, and impacts	
Risk-treatment Related Concepts		
Security requirements	Combination of flow objects connected by sequence flows	

Figure 5 depicts an example of a security risk-aware BPMN model. In this example a user inserts and submits data through an input interface. The padlock on the data object in the

input interface pool denotes that this data is confidential and should not be made readable for anyone else but the intended receiver. Therefore, a security requirement has been introduced that requests the data to be made unreadable for third parties before it is sent on its way. The server receives the data and employs it. The data object with a padlock in the server pool denotes, that the data confidentiality criteria has been accomplished.

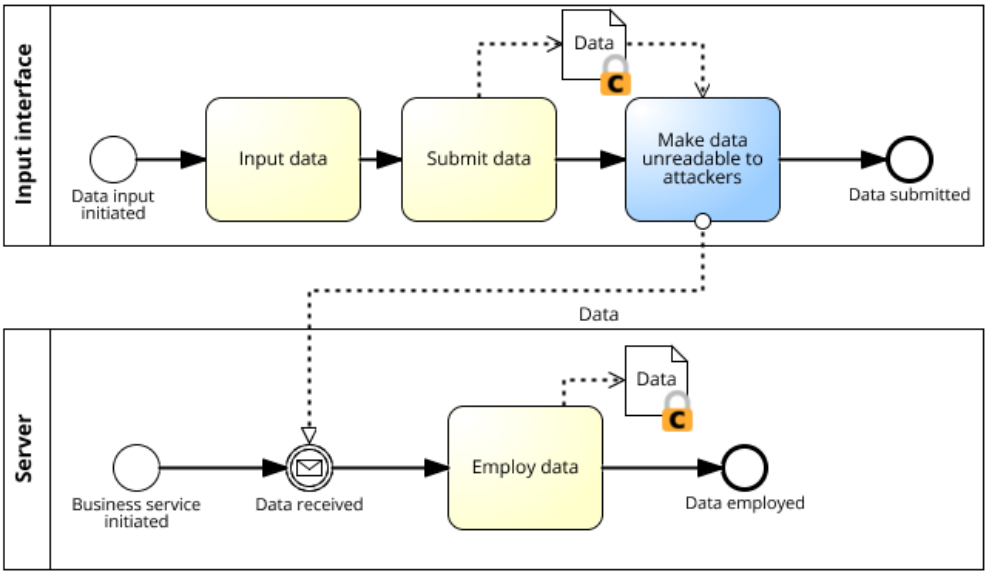


Figure 5. An Example of Security Risk-aware BPMN

The primary reason for choosing BPMN for presenting our contributions is that the existing body of work uses BPMN (Ahmed & Matulevičius, 2013). Thus for consistency purposes it makes sense to continue the expansion of the existing work using the established approach.

2.2.2 Alternative Modelling Languages for Security Risk Management

Mal-Activity Diagrams (MAD) are an extension of the Unified Modelling Language (UML) activity diagrams. Their main objective is to describe business processes, workflows and procedural logic (Chowdhury *et al.*, 2012). The common workflow for designing a Mal-Activity Diagram begins with creating a regular activity diagram and adding malicious activities, swim lanes and decisions to it. A limitation of MAD is an inability to depict security concepts like vulnerabilities, events, risks and also security criteria.

Misuse cases are extensions of use cases for supporting security requirements (Soomro & Ahmed, 2012). While use cases depict desirable functionalities in the IS, misuse cases depict functionalities that the system should not allow. A mis-actor is an initiator of misuse cases. Thus it is an actor whom the system should not support. Mis-actors and misuse cases are depicted in inverted colours in misuse case diagrams, as opposed to actors and use cases. Misuse case diagrams also support additional relations such as "detects" and "prevents" for indicating measures that detect and prevent unwanted activities. However, misuse cases are unable to distinguish between some ISSRM concepts such as IS assets, business assets and security requirements (Matulevičius *et al.*, 2008). Another limitation of misuse cases is lack of modelling constructs for concepts like security criteria, risk and impact.

Secure Tropos is an extension of Tropos and *i** to provide support for security requirements modelling. It extends the two languages' concepts and processes while integrating techniques like security reference diagrams and security attack scenarios. It can be applied for early and late requirements engineering and architectural and detailed design (Matulevičius,

2012). Secure Tropos addresses security requirements simultaneously with other IS requirements. Additionally, social aspects of security are addressed at an early stage. Secure Tropos applies multiple models for security analysis. There is the security enhanced actor model (SEAM) which is intended for identification and analysis of IS actors and dependencies between them. The model describes security constraints that the actors must respect. For example, in case of a secure dependency constraint, a dependee must respect the security constraints expected by a depender. In addition to SEAM, there is also the security enhanced goal model (SEGM), which describes the reasoning behind each actor's goals, plans and resources. Finally, there is the security reference diagram that is intended for identification of goals behind possible attacks.

2.2.3 Comparison of Modelling Languages for ISSRM

In a study performed by (Matulevičius, 2012), all of the four aforementioned modelling languages' semiotic clarities for presenting ISSRM concepts were analysed. Table 3 summarises the semiotic clarities of each language.

There are only a few one-to-one correspondences between ISSRM concepts and language semantics. All languages have overloaded semantics for describing assets, meaning the language constructs used for describing assets have multiple meanings. All languages also have redundancies which means that there are overlapping semantics between two or more language constructs. Incompleteness is another issue in all of the modelling languages, meaning it is not possible to convey sufficient information about some ISSRM concepts. All of the languages also suffer from under-definition, meaning there are no dedicated semantics for presenting some ISSRM concepts.

As we can see, none of the languages completely correspond to the ISSRM domain model. This is understandable however, because none of the languages were designed specifically for security risk management in the first place.

Table 3. Semiotic Clarity of Modelling Languages, adapted from (Matulevičius, 2012)

Semiotic clarity	BPMN	Secure Tropos	Misuse cases	Mal-activity diagrams
One-to-one correspondence	Threat agent	Threat agent	Security criterion, impact, vulnerability, threat agent	Impact, threat agent, control
Redundancy	Assets	Event	Assets	Assets, attack method
Overload	Assets	Assets	Assets	Assets
Incompleteness	Security criterion, risk, impact, event, vulnerability, threat, risk treatment, control	Risk, impact, vulnerability, threat, risk treatment, control	Risk, event, threat, risk treatment, control	Security criterion, risk, event, vulnerability, threat, risk treatment
Under-definition	Assets, attack method, security requirements	Assets, security criterion, attack method, security requirements	Assets, attack method, security requirements	Assets, security requirements

2.3 Security Requirements Engineering Using Patterns

In this chapter we examine the security requirements engineering methodology that enables us to elicit security requirements using patterns. We also give descriptions of some alternative popular security requirements engineering approaches.

2.3.1 Security Requirements Elicitation from Business Processes (SREBP)

SREBP is a security requirements elicitation method developed at University of Tartu (Ahmed, 2014). It is based on the ISSRM domain model and security risk-oriented patterns (Ahmed, 2014). The method is composed of two sequential stages (see Figure 6). During the first stage business assets are identified using value chains and business process models as an input. Security objectives are then determined for each asset.

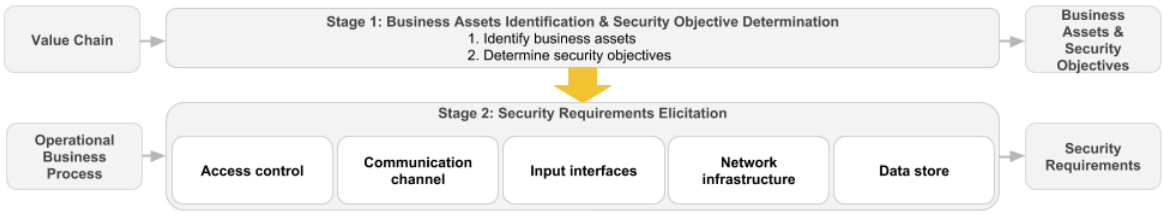


Figure 6. Security Requirement Elicitation from Business Processes (SREBP), adapted from (Ahmed, 2014)

At the second stage, security requirements are elicited using operational business processes. During this stage, the information system is approached at five contextual areas: access control, communication channel, input interfaces, network infrastructure, and data store. Appropriate security risk-oriented patterns are then selected that mitigate relevant security risks in each of the contextual areas. We give a short overview of the five contextual areas below.

Access control contextual area addresses concerns related to access control policy to ensure the integrity and confidentiality of business assets (e.g., data). A proposed security risk mitigation method could be the implementation of a Role-based Access Control (RBAC) model. Permissions are granted based on existing job functions (i.e. roles) in the organization.

Communication channel is concerned with data exchange between business entities. The focus is on confidentiality and integrity of the data being transferred using a transmission medium.

Input interfaces covers the treatment of input data prior to processing it. This includes checks for compatibility (e.g., data format), completeness (e.g., fulfilment of mandatory fields) and logical errors (e.g., correctness of dates).

Network infrastructure is focused on the protection of network infrastructures used between different business entities to perform business operations. The goal is to ensure the availability of business operations derived from the operational business process models.

Data store contextual area is concerned with the protection of data during storage and retrieval operations in associated databases.

There are relevant security risk-oriented patterns that correspond to each contextual area. The patterns address potential risks and provide security requirements together with reasoning behind them. We discuss security risk-oriented patterns in more detail in chapters 3 and 4.

2.3.2 Alternative Security Requirements Engineering Approaches

There are numerous alternative methodologies available for security requirements engineering. Some popular examples are Security Quality Requirements Engineering (SQUARE, Mead et al., 2005), and Security Requirements Engineering Process (SREP, Mellado *et al.*, 2007). We give brief overviews of these methodologies below.

SQUARE. SQUARE is a security requirements elicitation, categorisation and prioritisation method originally developed at Carnegie Mellon University (Mead *et al.*, 2005). One of the main goals of the SQUARE methodology is to integrate security engineering into early stages of the software development process. The method consists of nine steps. We give brief overviews of each of the steps below.

Agreement on definitions. SQUARE is dependent on the participation of requirements engineers with security expertise and also representatives from the stakeholders. An agreement is formed on the security definitions used throughout the process. Definitions do not usually need to be invented, but can be obtained from sources such as IEEE (Electrical and Electronics Engineers).

Identification of assets and security goals. Once the requirements team and stakeholders have agreed on the definitions, work can begin on the identification of assets and security goals. A wide representative selection of stakeholders should be engaged to elicit all the various goals that different stakeholders have. After the assets and security goals have been identified, they also need to be prioritized based on their significance in the organization's business activity.

Development of artefacts to support security requirements definition. Artefacts in the form of threat scenarios, misuse or abuse cases, etc. are developed. This is to ensure that the organization as a whole has the same understanding of the inputs for requirements elicitation and to keep any assumptions at a minimum. These artefacts also serve as the input for risk assessment.

Performance of risk assessment. With the help of a risk expert, an appropriate risk assessment method is selected for the project. Risk assessment provides an overview of the severeness of different security exposures.

Selection of elicitation techniques. The team chooses an elicitation technique that would be most effective for gathering the particular stakeholders' security requirements needs. After this, the actual security requirements elicitation can start.

Elicitation of security requirements. Using the artefacts developed earlier, initial requirements are developed.

Categorization of security requirements. The requirements are categorized to determine whether they are essential requirements, goals or architectural constraints. Categorized requirements also serve as the input for the requirements prioritization step.

Prioritization of requirements. SQUARE does not specify a prioritization technique. The technique should be selected based on the specific project's characteristics. Consequences of a security breach can be considered for prioritization as well as the cost/benefit ratio of satisfying different requirements.

Inspection of requirements. As a final step, the prioritized list of requirements are inspected using methods such as Fagan or peer reviews. The final output is a documentation of security requirements that fulfils the security goals of the project.

SREP. SREP is an asset-based and risk-driven security requirements engineering method that integrates Common Criteria (CC) into iterative and incremental software development models (Mellado *et al.*, 2007). SREP makes use of a Security Resources Repository (SRR) to enable reuse of security requirements, assets, threats and countermeasures in future development iterations. The core of SREP consists of nine activities which should be carried out in each development iteration throughout the incremental development process. The activities are the following:

Agree on definitions. During this activity the organization determines the stakeholders and forms an agreement on the security definitions, organizational security policies and the security vision of the IS. The security definitions are preferably taken from the ISO/IEC and IEEE standards. All of this shall be documented in the Security Vision Document.

Identify vulnerable and/or critical assets. The goal of this activity is to have a collective understanding of the assets that are important to the system. The assets are determined for example by analysing functional requirements or interviewing the stakeholders, but also by surveying the Security Resources Repository (SRR).

*Identify security objectives and dependencies. Security o*bjectives are established for each of the assets identified during the previous activity. SRR can also be consulted for this, provided it contains the assets identified in the previous activity with their associated security objectives. Otherwise the objectives are established taking into account the previously agreed upon security policy as well as legal requirements and constraints. Following this, dependencies are established between the security objectives. As a final output of this activity, the objectives and dependencies are documented in the Security Objectives Document.

Identify threats and develop artefacts. The goal of this activity is to identify all of the threats that compromise the identified assets. If threats are not already available in the SRR, artefacts such as misuse cases, attack tree diagrams etc. are developed. Also, public domain sources should be searched to identify potential vulnerabilities in the IS. As a result, the Security Problem Definition Document is generated which contains the identified threats, assumptions and conformance claims.

Risk assessment. This activity takes the previously compiled threat list as an input. For each threat, its probability, potential impact and risk are determined. The results are captured in the Risk Assessment Document.

Elicit security requirements. Suitable security requirements are elicited that mitigate the threats sufficiently based on the risk assessment results. The Security Requirements Specification Document is created as a result of this activity.

Categorize and prioritize requirements. Requirements are categorized and prioritized according to their importance, i.e., the severeness of the risk they mitigate.

Requirements inspection. The quality of the output from the previous activities is evaluated by reviewing all the previously generated artefacts (i.e. documents, models, requirements) and the results are presented in a Validation Report. Security requirements conformance to the IEEE 830-1998 standard is checked.

Repository improvement involves extending the SRR with new elements such as assets, threats, and requirements etc. that were identified during the previous eight activities. This information can be reused in future iterations. As the project development matures, the quality of the requirements stored in SRR is likely to increase because any inconsistencies, mistakes or ambiguities are more likely to be discovered.

2.4 Summary

Chapter 2 presented security related concepts such as security engineering and security risk management. Several security engineering and security risk management methods were explored. Additionally, model driven approach for security risk management was studied including modelling languages used for security risk management. The next chapter goes into more detail about security risk-oriented patterns and security patterns in general.

3 Security Patterns

This chapter gives an overview of security patterns, categories of security patterns, and threat patterns. There exist numerous classification systems for categorizing security patterns. In subchapter 3.1 we introduce one of the more commonly referenced classification systems by (Schumacher, 2003). We are also aware that there are numerous resources available for threat patterns (e.g., Mitre's CAPEC, Microsoft's STRIDE). However, in this thesis we would like to focus on the security threats taxonomy for distributed systems by (Uzunov & Fernandez, 2013). We discuss these threat patterns in subchapter 3.2. We also discuss the library of security risk-oriented patterns (SRPs) that SREBP is based on in subchapter 3.3. As a preparation for our contributions, we align the SRPs with the threat patterns by Uzunov & Fernandez (2013) to see which threat patterns the SRPs correspond to.

3.1 What are Security Patterns?

"A security pattern describes a particular recurring security problem that arises in a specific security context and presents a well-proven generic scheme for a security solution" (Schumacher, 2003). Although patterns can be used in other areas of software development, in this paper we specifically focus on security patterns.

Software projects tend to run into similar problems. Often these problems do not require new tailor-made solutions, but can be solved with solutions that have already been successfully applied in previous situations. Novice software engineers however lack this practical experience to rely on. This is where patterns come in handy. Instead of spending time and resources on working out new solutions, software developers can opt to implement already proven solutions by applying the appropriate patterns.

The objective of a pattern is to represent a high-quality and proven solution that solves a given problem optimally. A pattern does not aim to be new and innovative, but rather rely on a solution that has already been successfully applied in the past. In addition to that, a pattern ought to be as independent as possible from specific implementation technologies and tools. Not any solution to a problem qualifies as a pattern. In order for a solution to become a pattern it must first go through an organized peer review. Most patterns get their quality assurance at conferences like PLoP and EuroPLoP.

Patterns are not independent islands. They are part of a hierarchy where larger patterns contain smaller patterns that solve sub-problems of the main problem. Patterns can be combined together with other patterns and form a larger design. Because of this combinability, patterns can effectively be applied in complex and large-scale software systems.

The pattern catalogue introduced by (Schumacher *et al.*, 2005) divides patterns into classes. We give brief overviews of each of these classes below.

Enterprise Security and Risk Management patterns include policies, directives and constraints that apply to all systems in an enterprise. They mitigate risks that weigh on all parts of the enterprise and help resolve the security needs of the enterprise's assets.

Identification and Authentication patterns describe specific requirements for how users identify and authenticate themselves in the system. The designs make use of security measures such as passwords, biometrics, hardware tokens etc.

Access Control Model patterns concern authorization i.e. which resources are available to whom. This includes role-based access control where access is given based on daily tasks of users. They also address how access to sensitive data should be assigned.

System Access Control Architecture patterns are essential for systems that explicitly deny or permit the use of access control security services. The patterns deal with the architecture of the software system.

Operating System Access Control patterns concern how access to files in an operating system is controlled and how authorization is enforced on processes that wish to access a certain resource.

Accounting patterns prescribe how events involving the assets are tracked. This involves intrusion detection mechanisms that analyse the captured events for indications of security violations.

Firewall Architecture patterns are for controlling attacks on the network. They guide analysts to make a decision of trade-offs between complexity, speed and security.

Secure Internet Applications patterns deal with obscuring data and information about the surrounding environment, securing message channels, interaction methods with known partners and the location of the organization's web servers.

Cryptographic Key Management patterns are solely dedicated to secure communications. They prescribe cryptographic key generation and session and public key exchange methods.

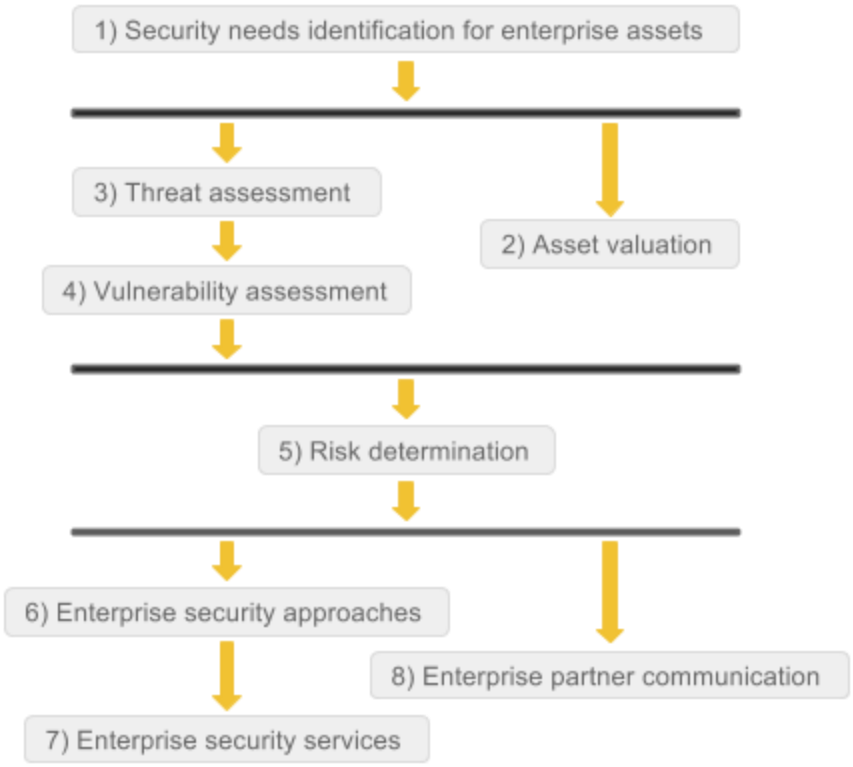


Figure 7. Sequence of Enterprise Security and Risk Management Patterns, adapted from (Schumacher *et al.*, 2005)

Figure 7 depicts the sequence of enterprise security and risk management patterns described by (Schumacher *et al.*, 2005). The sequence closely follows the ISSRM domain model (see Figure 4) as we shall now see.

Security needs identification for enterprise assets covers the security criterions class of the ISSRM domain model. It helps decide whether security is needed and if so, what the security criterions for the various enterprise assets are.

Asset valuation is closely related to IS asset and business asset classes in the ISSRM domain model. Valuation helps to determine the value of the enterprise's assets based on the costs of compromise (e.g. hard costs like fees and soft costs like loss of market share) on assets.

Threat assessment is connected to the threat class of ISSRM. Threats are about the likelihood of an attack occurring. Threat assessment intends to examine which threats are more likely to compromise the enterprise's assets and the likelihood of their occurrence.

Vulnerability assessment is related to ISSRM's vulnerability class. It addresses known weaknesses of the information system's assets and the severity of each vulnerability if it were to be exploited by an attacker.

Risk determination relates to event and risk classes of ISSRM and it guides us in assessing each risk's significance based on asset valuation, threat assessment and vulnerability assessment. The risks can be prioritized based on their significance levels.

Enterprise security approaches represent ISSRM's security requirement class which determines the prevention, detection and response techniques employed for protecting the enterprise's assets from appropriate risks.

Enterprise security services presents the services for protecting the enterprise's assets based on the previously established security approaches. It is related to the control class of ISSRM. Examples of such requirements include authentication, identification, authorization etc.

3.2 Threat Patterns for Distributed Systems

Uzunov & Fernandez (2013) divide security threats to distributed systems into eight categories and meta-security threats to the security infrastructure of systems into four categories. The security threat categories are based on the nature of the threats. Table 4 contains the complete list of first level security threat patterns and Table 5 contains the list of second level (i.e., meta-security) threat patterns, divided by category. We describe each threat category in more detail below.

Identity attacks are attacks where an attacker fabricates a new identity or claims to own an existing identity in the system. This allows the attacker to gain special privileges in the system and access sensitive data.

Network communication attacks target communications between different components of the system. There are many threat patterns in this class of attacks. An example threat pattern is message secrecy violation where an attacker intercepts the communications between endpoints and obtains the transmitted contents. Message integrity violation is an extension of the aforementioned threat in which the attacker actually manipulates with the data being transmitted, whether by corrupting, deleting or replacing it.

Network protocol attacks are a special class of network communication attacks which specifically target the underlying protocols being used for message communication. This is another class with many threat patterns. One example of this class is protocol field modification where an attacker modifies a field of the protocol causing unwanted behaviour in the communications process. Another example is protocol initial state exploitation where the attacker exploits the protocol initiation procedure, enabling the attacker to initiate more connections than is allowed.

Passing illegal data is a class with just one threat pattern which is data injection. As the name indicates, the attacker sends malicious data to a target which then processes it. Examples of injected data are SQL statements, binary code, OS commands etc., depending on the target context.

Stored data attacks are targeted towards on-storage data. A threat pattern belonging to this class is data corruption in which the attacker harms the stored data, potentially causing instability in the target system.

Remote information inference threats aim to gather information from the target component. This class of threats includes patterns like scanning of systems for useful information (e.g. software versions), probing of systems for known vulnerabilities and disclosure of system output (e.g. error messages) information.

Loss of accountability threats deal with altering attributes concerning accountability. The threat patterns in this threat class are track erasing where the attacker manipulates with auditing information and repudiation where a user negates being the executor of some operation in the system.

Uncontrolled operations is a class of threats aimed at exploiting system functionality. This is another class with many threat patterns. A threat pattern belonging to this class is unauthorized access where a user gains access to the system's resources without authorization. Another pattern is resource exhaustion where the system's resources are used excessively causing delays in system responsiveness.

The four meta-security threat categories with their respective descriptions are as follows.

Cryptography attacks are aimed at countermeasures which apply cryptography. One example of a threat pattern in this class is abuse of a weak cryptography algorithm which might allow the attacker to read sensitive data. Another form of attacks are password attacks. The easiest targets are passwords passed in plaintext and password storages that do not apply salting of password hashes.

Countermeasure design attacks try to exploit mistakes in the countermeasure setup. This class of attacks contains three patterns which are use of default credentials, bypassing controls and leveraging authorization model. Use of default credentials is self-explanatory – the attacker uses default credentials of a software for authentication. Bypassing of controls is a potential threat in systems that have insecure access points. Leveraging of authorization models threats systems which fail to enforce authorization correctly.

Configuration and administration attacks exploit configuration and administration weak points in a security infrastructure. Threat patterns belonging to this class are exploitation of bad policies and unauthorized modification of rights. Bad or missing policies might for example give users more control in the system than is necessary. Unauthorized modification of rights is a threat that affects systems which allow administrators to modify policies which they should be allowed to.

Network protocol attacks which we saw in the security threat categories also belong to the meta-security threats.

Table 4. First Level Security Threat Patterns, adapted from (Uzunov & Fernandez, 2013)

First level (security) threat patterns	
1. Identity attacks	UFP 1.1 Identity spoofing
	UFP 1.2 Advantageous identity allocation
2. Network communications attacks	UFP 2.1 Message secrecy violation (passive eavesdropping, reading plaintext message)
	UFP 2.2 Message integrity violation (active eavesdropping, modification)
	UFP 2.3 Message authenticity violation
	UFP 2.4 Traffic analysis, protocol sniffing
	UFP 2.5 Covert network channel
	UFP 2.6 Session hijacking
	UFP 2.7 Session state poisoning
	UFP 2.8 Route poisoning
	UFP 2.9 Message flooding
3. Network protocol attacks	UFP 3.1 Message replay
	UFP 3.2 Message reuse
	UFP 3.3 Protocol field modification
	UFP 3.4 Use of abnormal packet sizes
	UFP 3.5 Use of abnormal packet sequencing (reordering)
	UFP 3.6 Use of reserved protocol packets
	UFP 3.7 Protocol initial/end state exploitation
4. Passing illegal data	UFP 4.1 Injection
5. Stored data attacks	UFP 5.1 Corruption
6. Remote information inference	UFP 6.1 Scanning (information gathering)
	UFP 6.2 Probing (vulnerability checking)
	UFP 6.3 Output information disclosure
	UFP 6.4 Data inference
7. Loss of accountability	UFP 7.1 Track erasing
	UFP 7.2 Repudiation
8. Uncontrolled operations	UFP 8.1 Unauthorized access
	UFP 8.2 Invoking unauthorized operations
	UFP 8.3 Spoofing privileged processes (transitive actions)
	UFP 8.4 Unsafe code execution
	UFP 8.5 Exploitation of tight component coupling
	UFP 8.6 Process overflow attack (buffer/integer overflows)
	UFP 8.7 Exploiting concurrency flaws
	UFP 8.8 Resource exhaustion
	UFP 8.9 Targeted process crashing

Table 5. Second Level Security Threat Patterns, adapted from (Uzunov & Fernandez, 2013)

Second level (meta-security) threat patterns	
9. Cryptography attacks	UFP 9.1 Forging cryptographic credentials
	UFP 9.2 Abuse of weak algorithm
	UFP 9.3 Exploiting vulnerable security protocol
	UFP 9.4 Password attacks (guessing, brute force, rainbow tables, etc.)
10. Countermeasure design	UFP 10.1 Use of default credentials
	UFP 10.2 Bypassing controls
	UFP 10.3 Leveraging authorization model
11. Configuration/administration	UFP 11.1 Exploiting bad policies
	UFP 11.2 Unauthorized modification of rights (meta-authorization policies)

3.3 Security Risk-oriented Patterns

The library of security risk-oriented patterns compiled by (Ahmed & Matulevičius, 2013) contains five patterns. The patterns give an overview of the secured assets, risks and risk treatment methods. By applying these patterns to business processes, business analysts can design secure business processes on their own. We shall give brief descriptions of the security patterns below.

Security risk-oriented pattern (SRP) 1 secures data from unauthorized access. The security criteria for this pattern is confidentiality of the data stored on a business server. There is a risk that a user might request sensitive data from the server with the intention of misusing it. As a risk reduction measure the pattern proposes checking of access rights during data retrieval requests. Sensitivity levels must therefore be assigned to data and trust levels assigned to people or devices accessing the data as a measure for clearance verification.

SRP 2 ensures secure data transmission between business entities. Data confidentiality and integrity are two important security criteria for this pattern, because we do not wish to give access to the data to any third party and we wish that the data submitted by a client is the same data that is received by the server. However, data transmitted through a transmission medium faces at least two security vulnerabilities. A threat agent acting as a proxy could steal and read the data being transmitted. In addition, there is a risk of the transmitted data being modified during transmission. In order to reduce the risk of either of those attacks occurring, the pattern offers two security requirements. The data is encrypted in order to make the data unreadable to the threat agent and the data is also verified on the server to avoid the loss of data integrity.

SRP 3 ensures secure business activity after data submission. The security criteria for this pattern are availability and integrity of the business activity. Malicious scripts (e.g. SQL or xPath injections) submitted through an input interface could lead to disruption of the business activity, making the business activity unavailable and lose its integrity. What is more, execution of these malicious scripts on the server risks data confidentiality and integrity. To reduce the risk, the pattern proposes filtering of the incoming data which could be in the form of input validation, sanitisation, filtration and/or canonicalization.

SRP 4 secures business services against distributed denial of service (DDoS) attacks. The security criteria for this pattern is the availability of the business service. The risk is that there exists a threat agent with an access to many computers and the threat agent would exploit this access by making many simultaneous requests (e.g. TCP SYN or DNS flooding,

HTTP spidering etc.) at the target server, causing the business service to become unavailable for ordinary users. To reduce the risk of a successful DoS attack the pattern proposes a security requirement for checking for abnormal requests. Checking would involve filtering and classifying requests and detecting abnormalities with the requests. Requests classified as DoS attacks are discarded.

SRP 5 secures storage of data and data retrieval from storage. The security criteria for this pattern is confidentiality of data at the storage. There exists a risk that data could leak horizontally across the departments of an organization. A threat agent, in this case is a malicious insider with an access to the data could retrieve data from the storage. As a risk reduction measure the pattern proposes to make the data invisible using cryptographic or data protection techniques.

In the next subchapter we illustrate how different security threat patterns and security risk-oriented patterns are related to one another.

3.4 Alignment of Security Risk-oriented Patterns and Security Threat Patterns

Table 6 represents the alignment of security risk-oriented patterns to Uzunov & Fernandez's security threat patterns (2013). We can see how each security risk-oriented pattern is related to at least one threat pattern. Only seven security threat patterns currently have respective SRPs. However, the library of security threat patterns for distributed systems contains over 30 threat patterns. One of our goals in this thesis is to see which additional threat patterns should be mitigated by the SRPs. As another goal, we wish each SRP to address a single threat. Chapter 4 discusses these objectives in further detail and describes our efforts in working towards these goals.

Table 6. Alignment of Security Risk-oriented Patterns and Uzunov and Fernandez's Security Threat Patterns

Security Risk-oriented Pattern	Uzunov & Fernandez's Threat Pattern
SRP1. Secure data from unauthorized access	UFP 8.2 Invoking unauthorized operations
SRP2. Secure data transmission between business entities	UFP 2.1 Message secrecy violation UFP 2.2 Message integrity violation
SRP3. Secure business activity after data submission	UFP 4.1 Injection UFP 8.6 Process overflow attack
SRP4. Secure business services against denial of service attacks	UFP 2.9 Message flooding UFP 8.8 Resource exhaustion
SRP5. Secure storage of data and data retrieval from storage	UFP 5.1 Corruption UFP 7.1 Track erasing UFP 8.1 Unauthorized access

3.5 Summary

Chapter 3 was an introduction into the world of security patterns. We covered the definition, some important qualities of security patterns, and a classification system for security patterns. The library of security risk-oriented patterns was examined alongside the library of threat patterns for distributed systems. Links were pointed out between the existing security risk-oriented patterns and threat patterns. This chapter concludes the background part of this thesis.

4 Contribution

In this chapter we reclassify the security risk-oriented patterns (SRPs) based on the security threats they address in order to target individual security threats more intelligibly. Additionally, we expand the library of SRPs with extra security patterns in order to address security threats that are currently not considered. We also list the security threat patterns that are unsuitable for securing business processes and explain why. After revising the library of SRPs, we categorize the security patterns according to the five contextual areas of SREBP. Throughout this chapter we use security risk-aware BPMN to model the SRPs. Additionally, we present the SRPs in tabular form in appendix II. The tables contain additional information about the SRPs and explain in further depth how they correspond to the threat patterns.

4.1 Revision of Security Risk-oriented Patterns' Classifications

In chapter 3.4 we discussed the correspondence between SRPs and Uzunov & Fernandez's security threat patterns (2013). As we learned, four of the five SRPs mitigate more than one security threat. Ideally, a SRP should address a single security threat. This enables business analysts to only mitigate risks that are relevant in a system and prevents creating security requirements that target non-existent risks. Additionally, it allows each SRP to describe the security objectives, risks and risk treatment methods more explicitly. Therefore, we propose dividing the existing SRPs based on the threats they address. Table 7 represents the result of this division. We examine each case in further detail below.

Table 7. Division of SRPs Based on the Security Threats They Address

Ahmed & Matulevičius's SRPs	Divided SRPs
SRP1. Secure data from unauthorized access	SRP1a. Secure data from unauthorized access
	SRP1b. Secure business service from unauthorized access
SRP2. Secure data transmission between business entities	SRP2a. Secure data from secrecy violation during transmission between business entities
	SRP2b. Secure data from integrity violation during transmission between business entities
SRP3. Secure business activity after data submission	SRP3a. Secure business activity from injection attacks
	SRP3b. Secure business activity from buffer overflow attacks
SRP4. Secure business services against denial of service attacks	SRP4a. Secure business services against message flooding attacks
	SRP4b. Secure business services against resource exhaustion attacks
SRP5. Secure storage of data and data retrieval from storage	SRP5a. Secure data in data store from corruption by a malicious insider
	SRP5b. Secure the integrity of auditing information from a malicious insider
	SRP5c. Secure data in data store from unauthorized access by a malicious insider

SRP1 has been relabelled as SRP1a. It secures data from unauthorized access (see Table 19). The threat pattern that conforms to SRP1 (UFP 8.2 Invoking unauthorized operations) does not explicitly state the impact of the attack. Therefore, we have created a separate pattern labelled as SRP1b for securing business services from unauthorized access (see Table

20). The security requirements for SRP1b are the same as for SRP1a, which is to check the client's authorization rights before granting access to the business service (see Figure 32).

SRP2 has been divided into two separate patterns since it addresses two different threats during message transmission. The resulting SRP2a and SRP2b address threats to confidentiality and integrity security criteria separately and propose respective mitigations (see Table 21, Table 22). The security objectives, risks and requirements remain the same for both SRP2a (see Figure 33, Figure 34, Figure 35) and SRP2b (see Figure 36, Figure 37, Figure 38).

SRP3 has been relabelled as SRP3a (see Table 23). The security objectives (see Figure 39), risk (see Figure 40) and requirements (see Figure 41) remain the same. SRP3b is a new pattern that describes a special case of injection attacks where the attacker triggers a buffer overflow in the process execution environment (see Figure 43). This can result with control being passed to binary code injected by the attacker. The security objectives and requirements for both patterns are mostly the same with the only exception being that SRP2b recommends employing a whitelist of acceptable inputs as an additional security control wherever applicable. In case of buffer overflow attacks, the IS's security posture would also benefit from applying boundary checking (see Table 24).

SRP4 has been relabelled as SRP4a. SRP4a still focuses on distributed denial of service attacks (see Table 25) while SRP4b focuses on resource exhaustion attacks that lead to denial of service (see Table 26). Although the security objectives for both patterns are the same (i.e., availability of a business service), the risks and risk treatment methods are different. A resource exhaustion attack does not require a large volume of requests to be made at the target system like in case of distributed denial of service attacks. This attack rather relies on skilful manipulation of the target system. The best practice would be to eliminate any vulnerabilities (e.g., poor or lack of resource utilization restrictions) from the IS that could enable this attack in the first place. This can be achieved by implementing a throttling mechanism to limit the amount of resources available for allocation by a single user. A strong authentication and access control can also go a long way in order to mitigate this threat.

SRP5 has been relabelled as SRP5c (see Table 29). There are two additional patterns that are closely related to the malicious insider threat. SRP5a mitigates the threat of data corruption by an insider (see Table 27). This threat can be difficult to mitigate. Fortunately, there exist methodologies that are able to detect this attack using checksums (Barbará *et al.*, 2000). SRP5b is a special case of data integrity manipulation attack where the malicious insider attempts to modify or delete auditing information to erase tracks of malicious behaviour (see Table 28). Similarly to SRP5a, the security requirement is to implement a database integrity defence mechanism that can detect the attack.

4.2 Additional Security Risk-oriented Patterns

Table 8 presents a list of new security risk-oriented patterns that are based on Uzunov and Fernandez's security threat patterns (2013). We give an overview of each pattern by examining the secured assets, potential risks and security requirements.

SRP6 prevents leakage of confidential information (e.g., information about system configuration, credentials, private information) through error and standard response messages (see Table 30). Whenever an error condition emerges, it is usually desirable to return an explanatory output to the user in case there are any insufficiencies in the request made to the server. This scenario is illustrated in Figure 8. However, if these output messages are not properly filtered from confidential information, the attacker could employ this information to launch other attacks at the system, potentially causing unavailability of the business service as well

as loss of data integrity and confidentiality (see Figure 9). The security requirement for risk reduction is to filter confidential information from output messages. Figure 10 illustrates the introduction of the security requirement into a business process. The security requirement can be implemented by means such as generic error messages that do not leak information. Properly configuring server software so that debug messages are disabled in live environments should also be an obligatory task.

Table 8. Additional Security Risk-oriented Patterns

Security Risk-oriented Pattern	Threat Pattern for Distributed Systems
SRP6. Prevent harm to business service through leakage of information via error and standard response messages	UFP 6.3 Output information disclosure
SRP7. Protect business services from deadlock attacks	UFP 8.9 Targeted process crashing
SRP8a. Secure system from brute force password attacks	UFP 9.4 Password attacks (guessing, brute force, rainbow tables, etc.)
SRP8b. Secure system from unauthorized access through common or default usernames and passwords	UFP 10.1 Use of default credentials
SRP8c. Secure the system from account lockout attacks	Countermeasure design attack, no matching UFP

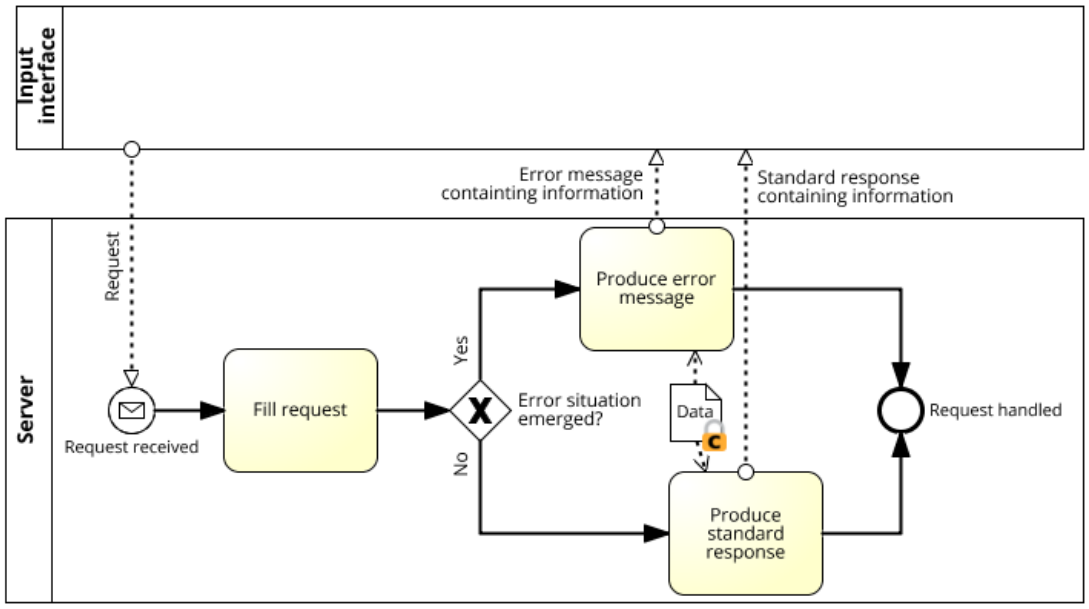


Figure 8. SRP6: Business Process Model with annotated security objectives

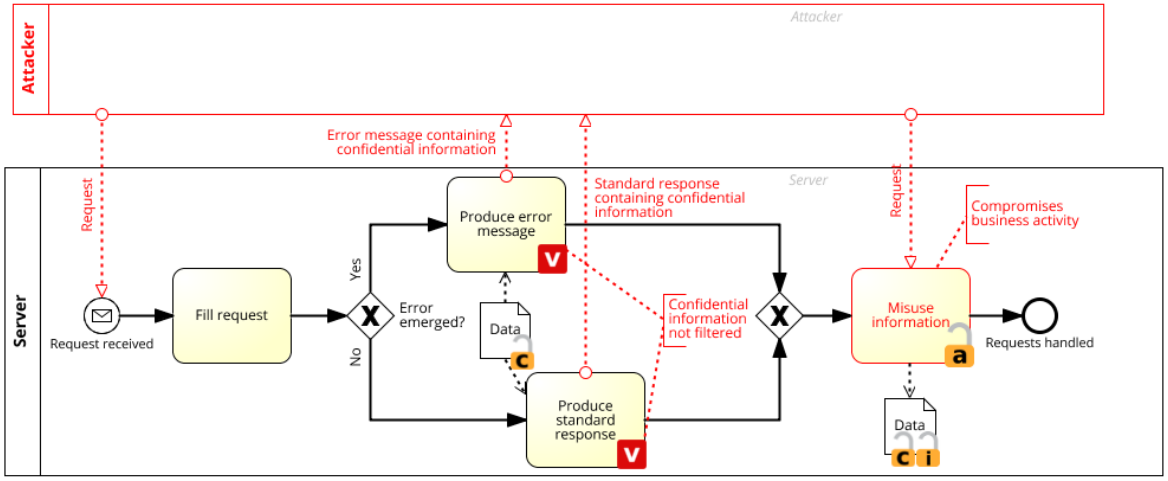


Figure 9. SRP6: Security Risk-oriented Business Process Model

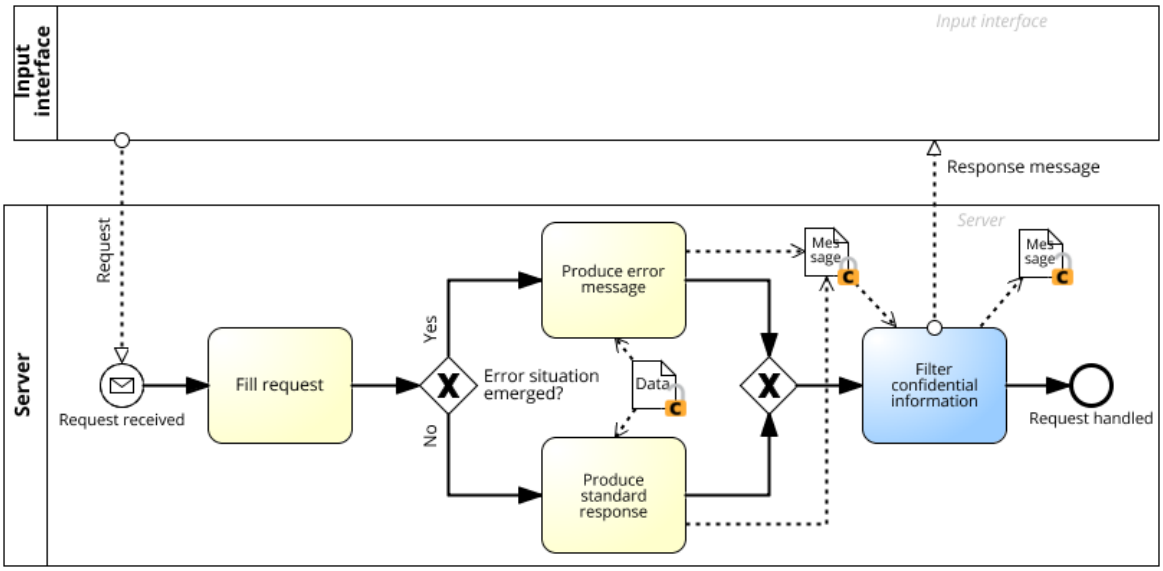


Figure 10. SRP6: Security Requirements for Business Process

SRP7 protects business services from deadlock attacks (see Table 31). A deadlock is a condition where multiple processes become dependent on one-another to finish and thus none of the processes ever finish. An attacker could skilfully trigger a deadlock condition and cause the business service to crash. Business availability is the security criterion in this pattern, as illustrated in Figure 11. The vulnerability that makes this attack possible is absence of process synchronization (see Figure 12). The security requirement proposed by SRP7 is therefore to synchronize execution and operation of simultaneous processes (see Figure 13). This requirement could be achieved for example by implementing non-blocking synchronization algorithms.

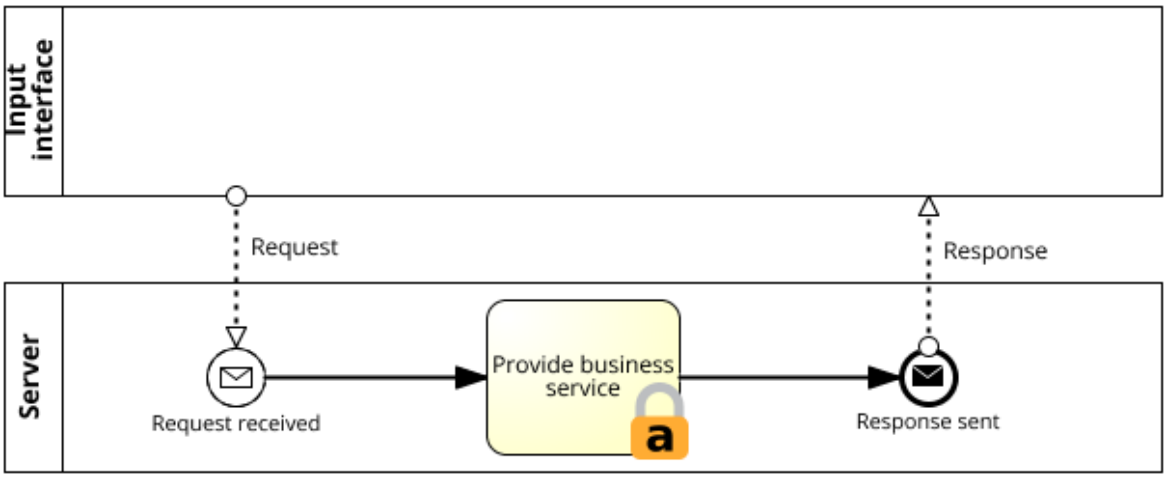


Figure 11. SRP7: Business Process Model with Security Objectives

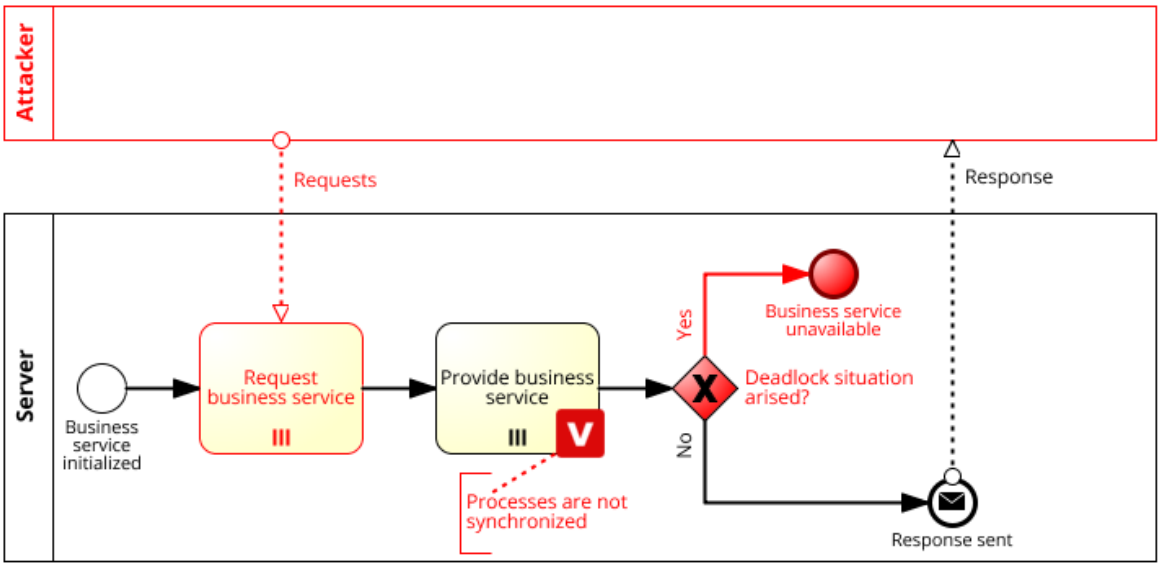


Figure 12. SRP7: Security Risk-oriented Business Process Model

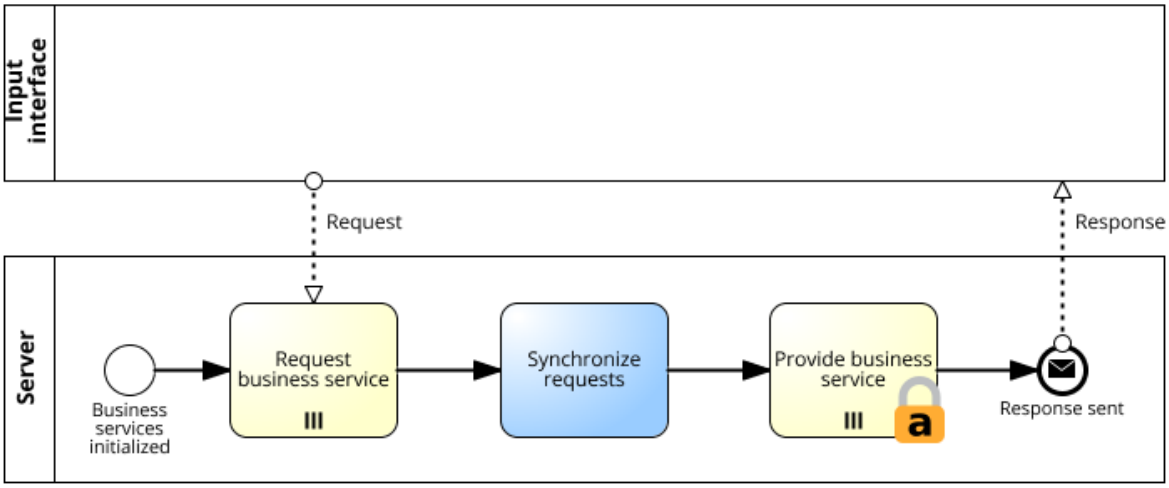


Figure 13. SRP7: Security Requirements for Business Process

SRP8a secures the system against brute force password attacks (see Table 32). A brute force attack is a trial-and-error method of trying every key from a key space until a legitimate key is found. The key space can be infinite or finite (i.e., a dictionary). The security criteria for this pattern are availability of the business service as well as integrity and confidentiality of data (see Figure 14). As illustrated in Figure 15, this threat is relevant in systems that make use of password authentication, but where failed login attempts threshold is not set. If a target user account were to have a weak password, then a brute force attack could be effective for gaining access to the system. The harm that a successful attack could lead to depends on the specific account's privileges. However, impact could range from loss of data confidentiality and integrity to unavailability of the business service. SRP8a introduces multiple security requirements to reduce this risk: implementation of strong password policy, invalid login throttling mechanism (see Figure 16), and mandatory periodical renewal of passwords.

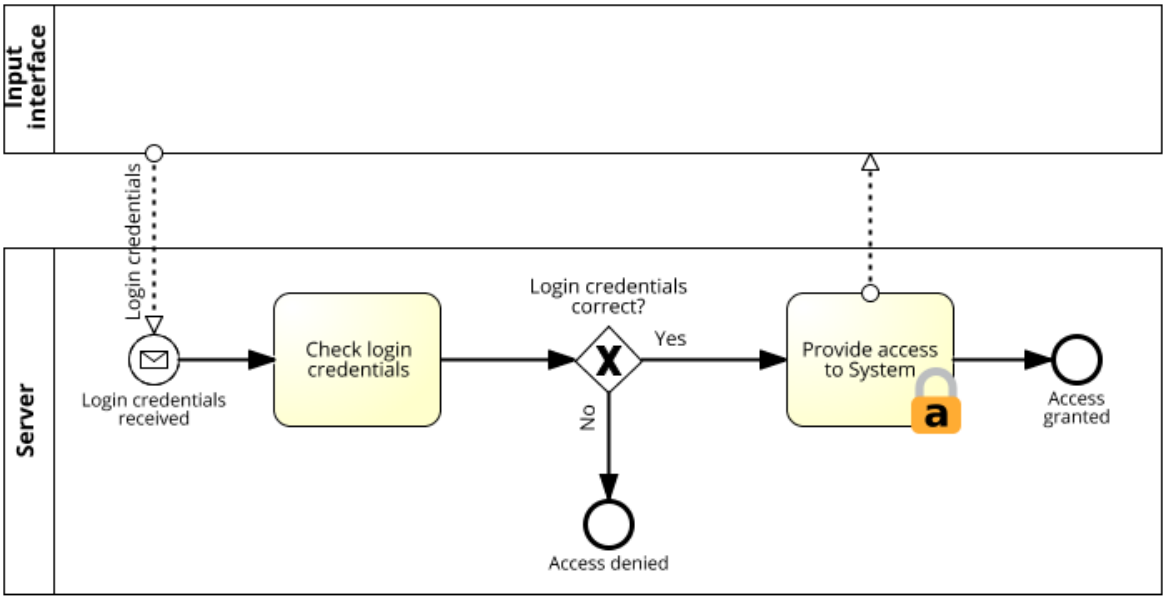


Figure 14. SRP8a: Business Process Model with Security Objectives

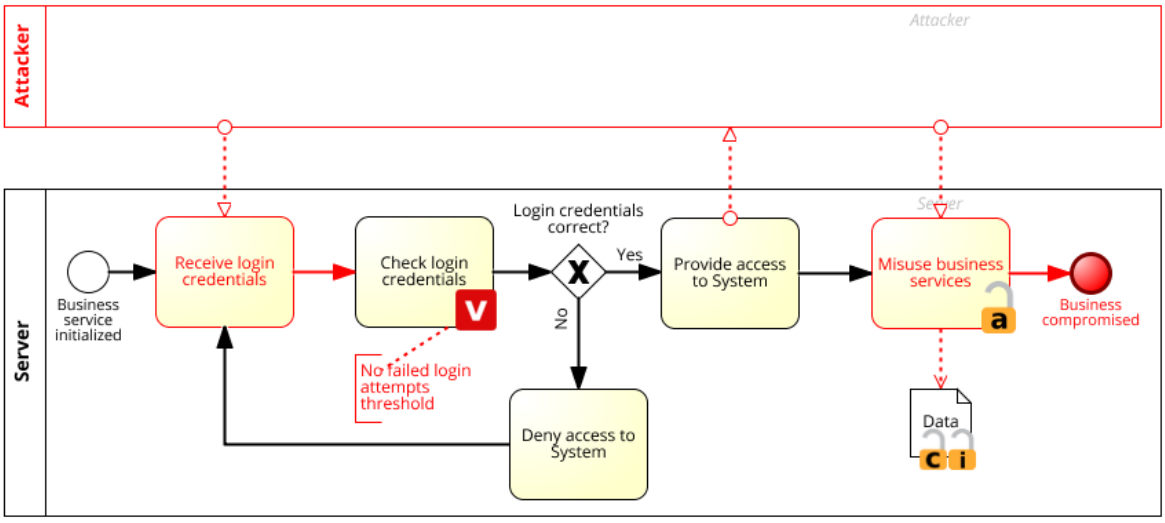


Figure 15. SRP8a: Security Risk-oriented Business Process Model

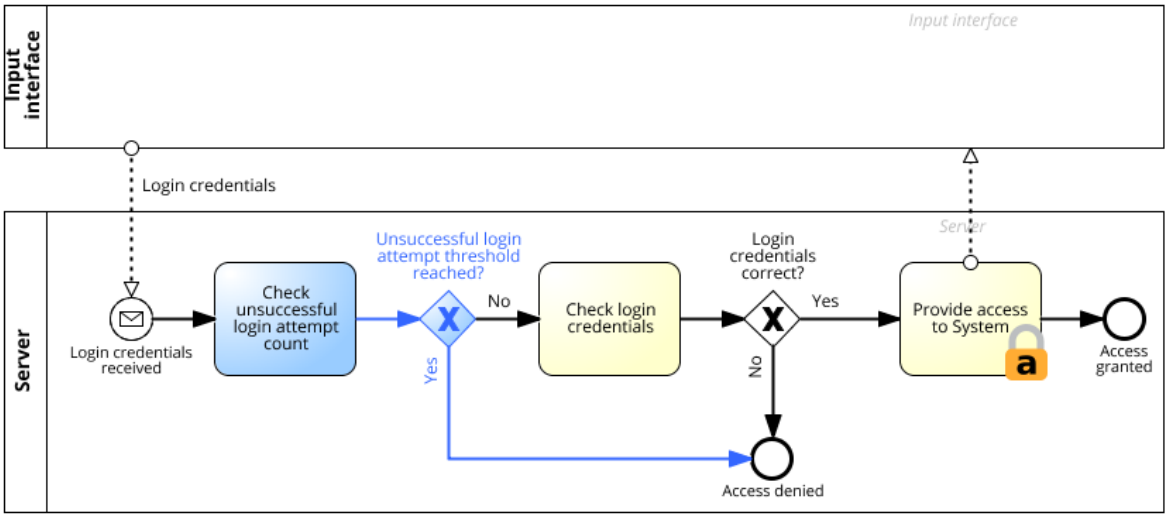


Figure 16. SRP8a: Security Requirements for Business Process

SRP8b is almost identical to SRP8a with the only distinction being that the key space in this pattern is a finite key space consisting of default user credentials (see Table 33). These credentials can for example be the ones used by developers during development to access the system and for some reason the credentials have made it into production. As a risk mitigation, default accounts should simply be removed from the system.

SRP8c is a sub-pattern of SRP8a that secures an IS from account lockout attacks (see Table 34). Although there is no matching security threat pattern for this attack, we have provided this SRP for completeness purposes. SRP8a proposes a countermeasure design that sets a threshold for invalid login attempts. This countermeasure could be exploited by an attacker to lock legitimate users out of the system, making the business services unavailable for them (see Figure 18). The security criterion for this pattern is availability of business service for all legitimate users (see Figure 17). To mitigate the risk of this attack succeeding, SRP8c proposes implementation of an account lockout attack detection mechanism (see Figure 19).

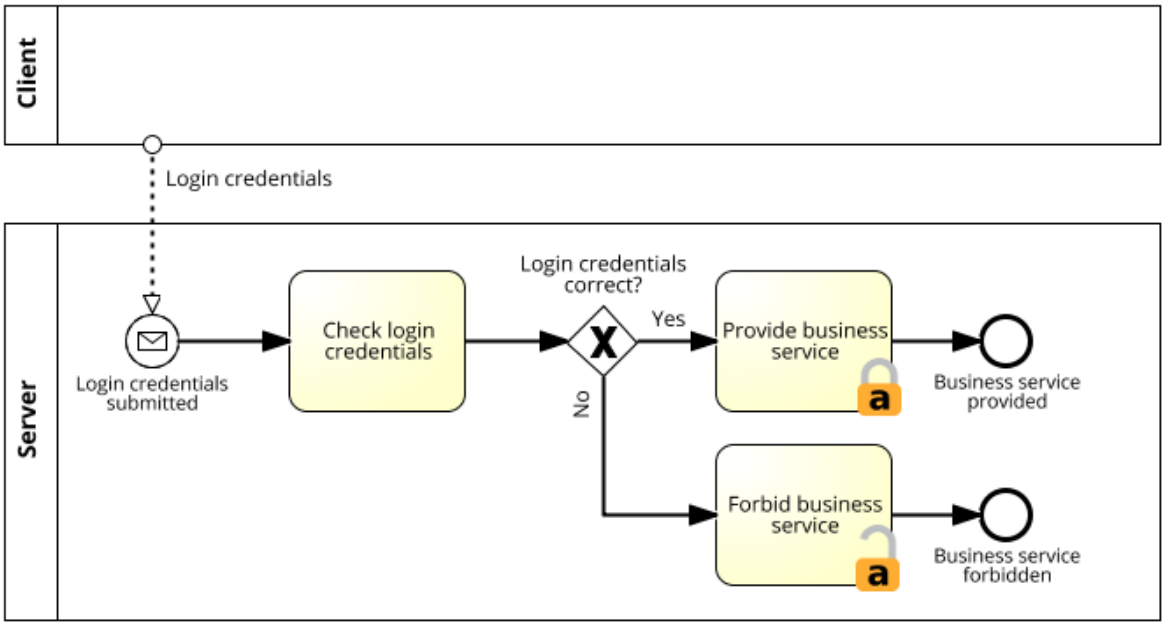


Figure 17. SRP8c: Business Process Model with Security Objectives

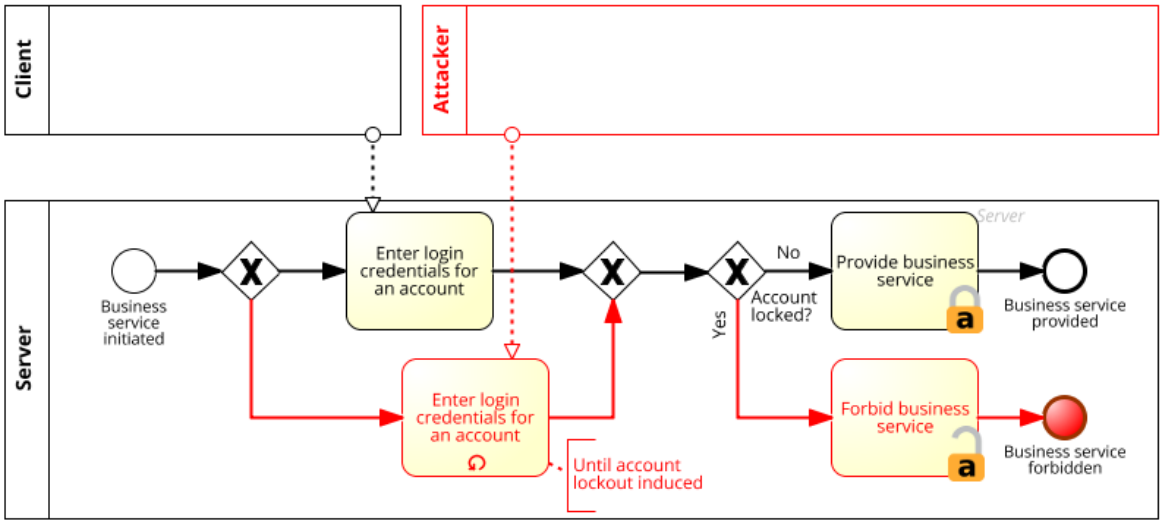


Figure 18. SRP8c: Security Risk-oriented Business Process Model

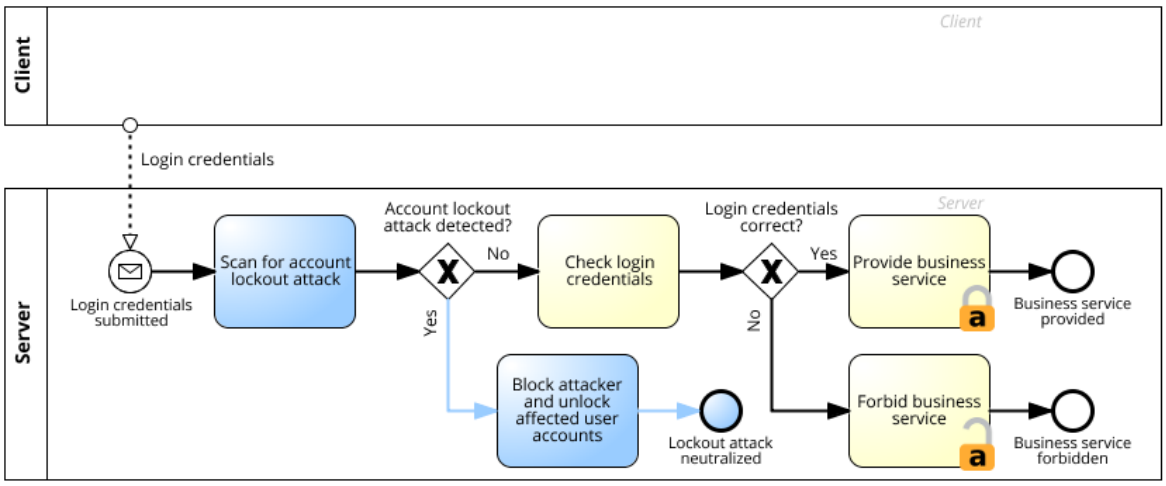


Figure 19. SRP8c: Security Requirements for Business Process

4.3 Alignment of Security Risk-oriented Patterns with SREBP

In chapter 2.3.1 we discussed the SREBP method in detail. However, we did not discuss how the security risk-oriented patterns fit into the five contextual areas. Figure 20 represents a revised illustration of the SREBP application method. We have categorized each SRP according to the contextual area it corresponds to. Thus, each contextual area contains SRPs that mitigate risks and elicit security requirements relevant to the specific contextual state.

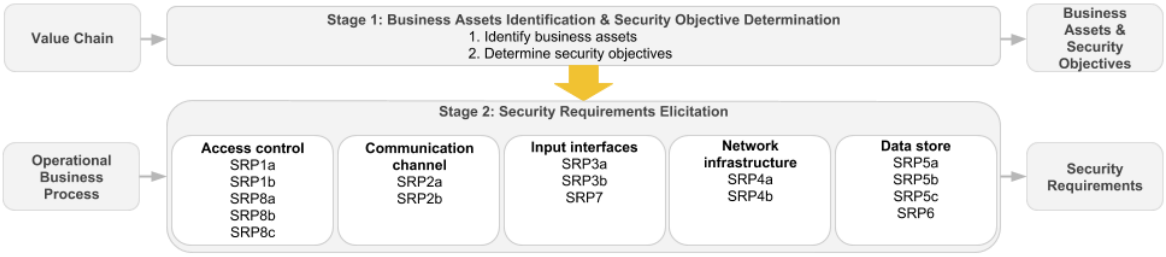


Figure 20. Illustration of SREBP Application Method

4.4 Uzunov & Fernandez's Threat Patterns Unsuitable for Securing Business Processes

Not all Uzunov and Fernandez's threat patterns (2013) qualify for a respective security risk-oriented pattern. In this subchapter we explain why some of the threat patterns are incompatible for securing business processes.

Identity attacks that are unsuitable for securing business processes are listed in Table 9. The descriptions given for these threats *do not describe an attack method*. Because of this, it is not possible to explicitly model the flow of this attack in a business process.

Table 9. Identity Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Identity attacks	UFP 1.1 Identity spoofing
	UFP 1.2 Advantageous identity allocation

Network communication attacks that are unsuitable for securing business processes are listed in Table 10. UFPs 2.3, 2.5, 2.6, and 2.8 *do not clearly describe the implementation of these attacks*. UFPs 2.4 and 2.7 on the other hand *do not explicitly specify the impact of these attacks*. We are not claiming these are not relevant threats, but rather that that it is difficult to conclude the impact of these attacks using the CIA triad.

Table 10. Network Communication Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Network communications attacks	UFP 2.3 Message authenticity violation
	UFP 2.4 Traffic analysis, protocol sniffing
	UFP 2.5 Covert network channel
	UFP 2.6 Session hijacking
	UFP 2.7 Session state poisoning
	UFP 2.8 Route poisoning

Network protocol attacks that are unsuitable for securing business processes are listed in Table 11. All of these patterns are *technical and low-level*. It is unlikely that most business analysts have comprehension about different network protocol types and purposes, not to mention protocol states and fields. Thus these threat patterns are unsuitable for mitigating security risks on a business process level.

Table 11. Network Protocol Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Network protocol attacks	UFP 3.1 Message replay
	UFP 3.2 Message reuse
	UFP 3.3 Protocol field modification
	UFP 3.4 Use of abnormal packet sizes
	UFP 3.5 Use of abnormal packet sequencing (reordering)
	UFP 3.6 Use of reserved protocol packets
	UFP 3.7 Protocol initial/end state exploitation

Remote information inference attacks that are unsuitable for securing business processes are listed in Table 12. Although the attack methods for these threat patterns are clearly described, *none of these threat patterns describe the impact of the attacks*. For example, it is

not clear which business assets are harmed and how by gathering information (e.g., type of operating system being used) about a system. We agree that this information may in some cases be beneficial to an attacker, if employed in a manner that violates some security criterion.

Table 12. Remote Information Inference Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Remote information inference	UFP 6.1 Scanning (information gathering)
	UFP 6.2 Probing (vulnerability checking)
	UFP 6.4 Data inference

There is only one accountability loss attack that is unsuitable for securing business processes (see Table 13). The description given for UFP 7.2 describes the impact, but *does not describe an attack method*. Therefore we cannot form a security risk-oriented pattern passed on this threat.

Table 13. Accountability Loss Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Loss of accountability	UFP 7.2 Repudiation

Uncontrolled operations attacks that are unsuitable for securing business processes are listed in Table 14. The descriptions of UFPs 8.3 and 8.5 *do not specify the impact of these attacks*. The description of UFP 8.4 on the other hand *does not explicitly state the attack method*. The TOCTOU attack that is described by UFP 8.7 is a *low-level systematic attack*, thus it does not fit well into a business process.

Table 14. Uncontrolled Operations Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Uncontrolled operations	UFP 8.3 Spoofing privileged processes (transitive actions)
	UFP 8.4 Unsafe code execution
	UFP 8.5 Exploitation of tight component coupling
	UFP 8.7 Exploiting concurrency flaws

Cryptography attacks that are unsuitable for securing business processes are listed in Table 15. These attacks target weaknesses in cryptographic measures (e.g., weak algorithms, vulnerable security protocols etc.). The best way to mitigate these attacks is to avoid using vulnerable cryptographic countermeasures in the first place. Also, another reason why these threats are irrelevant for us is that business analysts normally do not have the final word about which security control methods are implemented in order to satisfy the security requirements. This is a job best left for a security analyst.

Table 15. Cryptography Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Cryptography attacks	UFP 9.1 Forging cryptographic credentials
	UFP 9.2 Abuse of weak algorithm
	UFP 9.3 Exploiting vulnerable security protocol

Countermeasure design attacks that are unsuitable for securing business processes are listed in Table 16. Once again, these attacks have little to do with security requirements, because they are just poor choice or implementation of security controls.

Table 16. Countermeasure Design Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Countermeasure design	UFP 10.2 Bypassing controls
	UFP 10.3 Leveraging authorization model

Configuration/administration attacks (see Table 17) are also irrelevant for us, because *poor system configuration and administration cannot be* (and should not be) *mended by adding additional security requirements in a business process*.

Table 17. Configuration/administration Threat Patterns Unsuitable for Securing Business Processes

Threat category	Security threat pattern
Configuration/administration	UFP 11.1 Exploiting bad policies
	UFP 11.2 Unauthorized modification of rights (meta-authorization policies)

4.5 Summary

In this chapter we divided the SRPs up into numerous smaller patterns to explicitly express which security threats the patterns address. We also expanded the library of SRPs with additional patterns that target security risks which had not yet been mitigated. We gave explanations on why some security threats are not applicable for securing business processes. Finally, we explained how the SRPs fit into the five contextual areas of SREBP.

5 Validation

The thesis at hand explores ways to expand the library of security risk-oriented patterns to support the development of secure distributed system. In chapter 4 we proposed revisions to the security risk-oriented patterns library. The current chapter is dedicated to validation of our contributions to the library. It explains the validation method we applied, the validation results and potential threats to validity.

5.1 Validation Method

The research goal of this study is to understand what the applicability of the security risk-oriented patterns (SRPs) is for securing business processes in distributed systems. Figure 21 describes the validation method. We apply the revised security risk-oriented patterns to five business processes derived from real life scenarios in the aviation industry. These five processes were inspired by a turnaround process by (Nõukas, 2015). The processes are the following: *(i)* passenger check-in, *(ii)* baggage check-in, *(iii)* fuel service form issuing, *(iv)* fuel service form requesting and *(v)* loading instruction form requesting. Prior to engineering security requirements for the processes, their preciseness was verified by an expert from Tallinn University of Technology who has experience with business processes used in the airline industry. After application of the security risk-oriented patterns, we submitted the resulting security requirements to the same expert for evaluation.

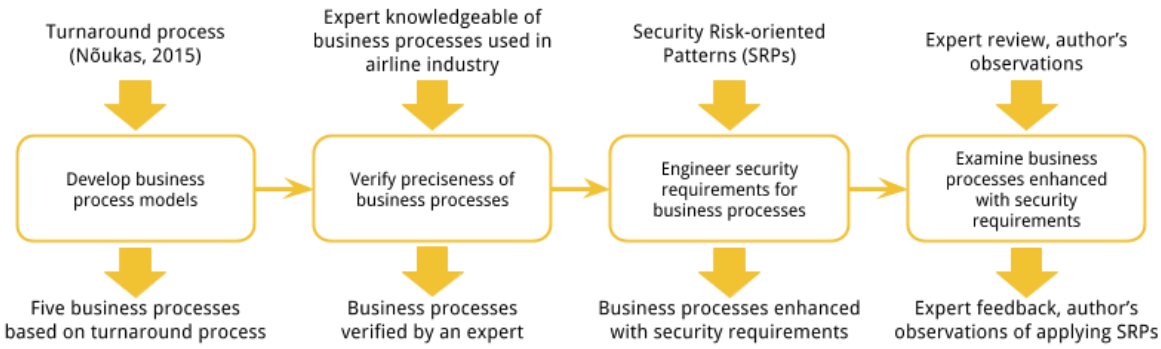


Figure 21. Flow of the Validation Process

In the following sub-chapter and in appendix III, we discuss the origination of each business process, present them using the BPMN modelling language and secure them by applying the necessary security risk-oriented patterns. We depict the security requirements for each process in security risk-aware BPMN models as well as in tabular form, where we also propose some potential control methods. For the passenger check-in process we apply each SRP separately and present the resulting risk and security requirements in separate models. This should help the reader better understand why each security requirement exists in the final security requirements model. For the other processes we only present a final model where all the security requirements are depicted.

5.2 Securing the Passenger Check-in Process

Figure 22 depicts a BPMN model of the passenger check-in process. This example scenario is based on airBaltic's online check-in process[1]. In this scenario, the passenger confirms their attendance on a flight via the Internet. The passenger enters their booking number and other required information (e.g., preferred seating, meal options etc.) and submits the data

[1] https://www.airbaltic.com/en/online-check-in-conditions

to the server. The server verifies the validity of the booking number. If the booking number is valid, the submitted information is stored in a data store and a boarding pass is issued to the passenger.

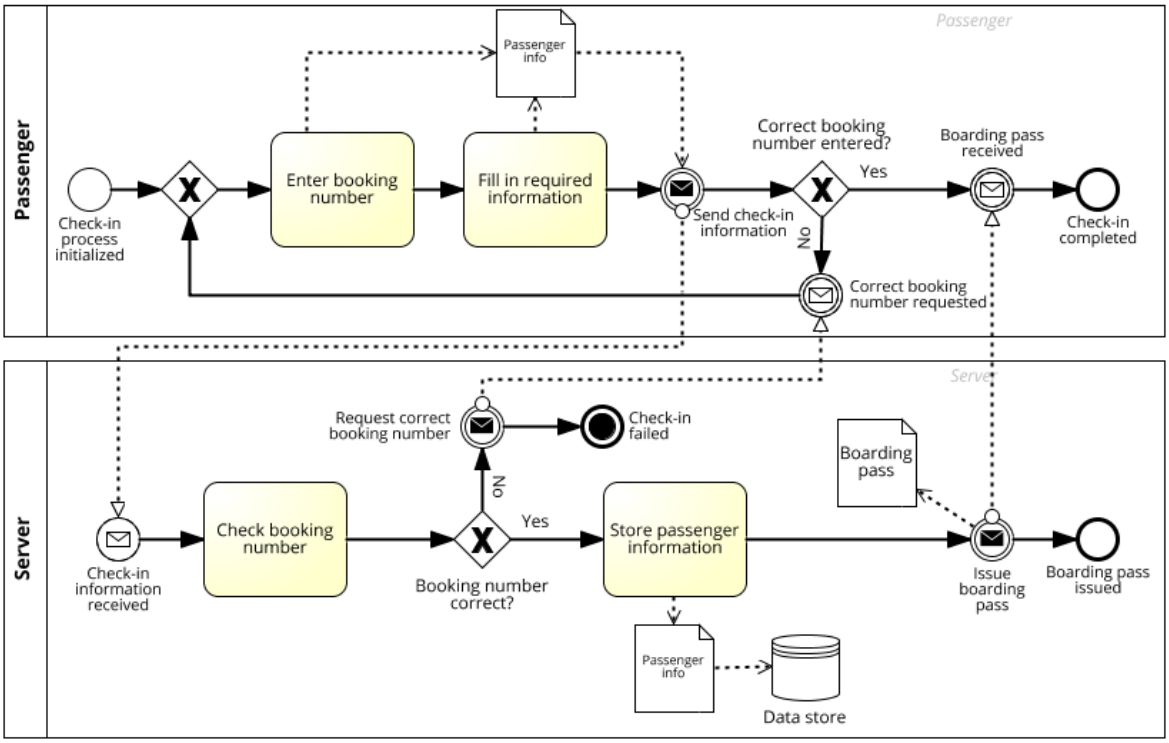


Figure 22. Passenger Check-in Process

We illustrate the application of SRP2a to the passenger check-in business process below. The models from application of the other SRPs are located in appendix III. Figure 23 and Figure 24 depict the risk analysis models from applying SRP2a. Figure 23 depicts a risk where an attacker intercepts the transmission between the passenger and the server by acting as a proxy. The attacker captures the passenger information being transmitted and keeps it for personal use. The impact of this attack is the loss of the passenger information's confidentiality.

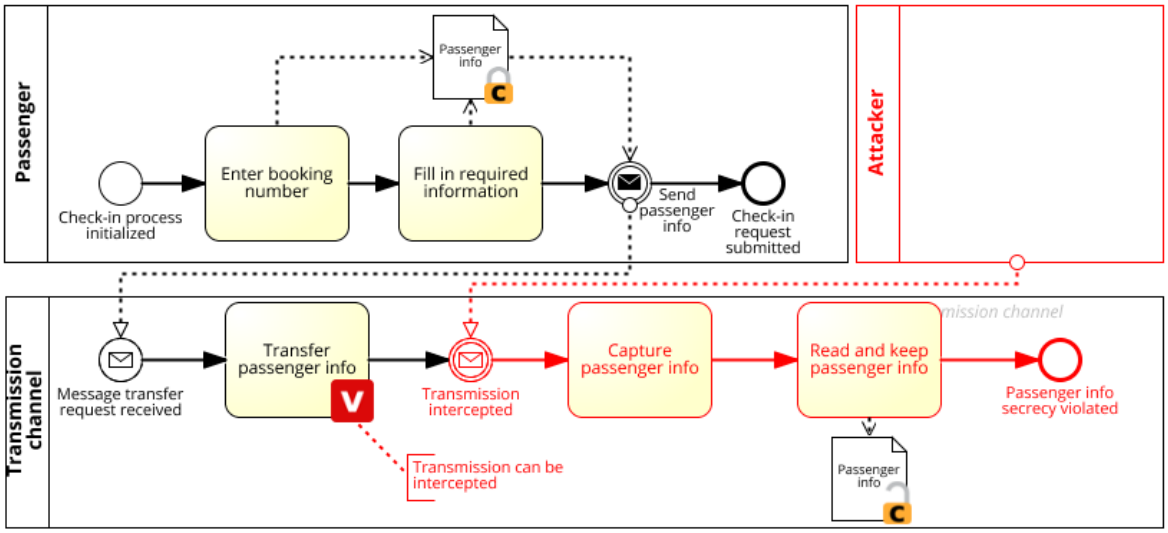


Figure 23. SRP2a: Security Risk-oriented Process Model (Business Asset: Passenger info)

Figure 24 addresses the same risk, but in this case the business asset is the boarding pass that is sent by the server to the passenger. An attacker again intercepts the transmission and this time captures the boarding pass. As a result, the boarding pass loses its confidentiality.

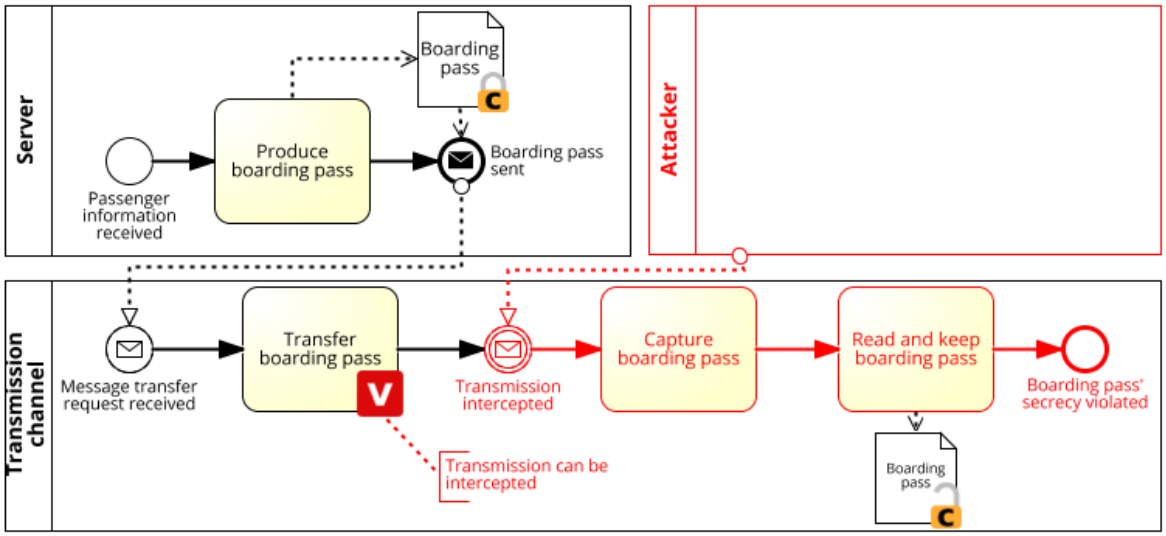


Figure 24. SRP2a: Security Risk-oriented Process Model (Business Asset: Boarding pass)

The security requirements for mitigating both of these risks are depicted on Figure 25. Before the passenger info is sent to the server, it is made unreadable for attackers (i.e., encrypted). Even if an attacker captures the encrypted information, the confidentiality of the passenger information remains intact. Similarly, before the server sends the boarding pass to the passenger, the boarding pass is made unreadable for attackers. These requirements ensure the confidentiality of both, the passenger information and the boarding pass during the transmission process.

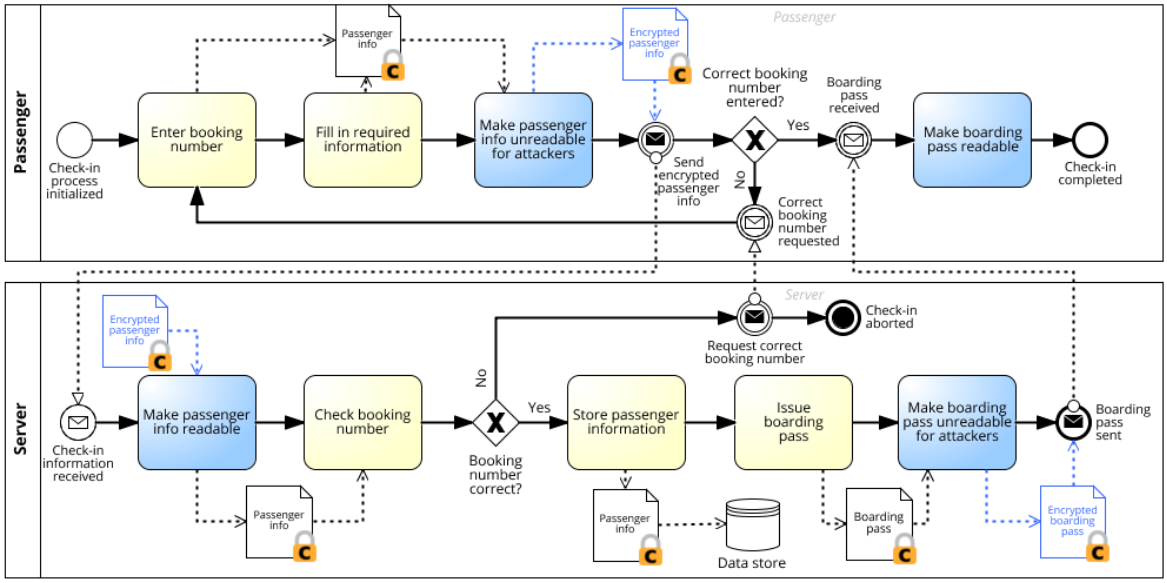


Figure 25. SRP2a: Security Requirements for Business Process

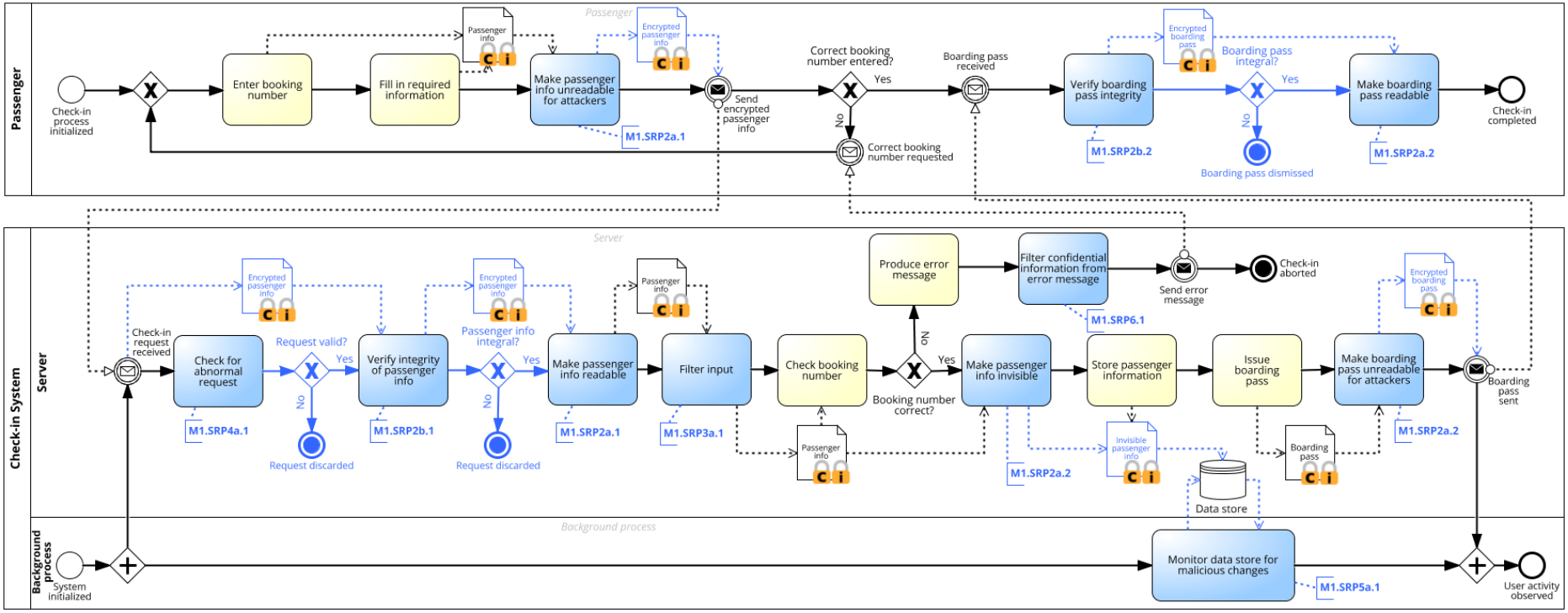


Figure 26. Security Requirements for Passenger Check-in Process

Figure 26 presents a BPMN model of the final security requirements for the passenger check-in process. Table 18 contains the security requirements for the passenger check-in process together with respective control methods.

Table 18. Security Requirements for Passenger Check-in Process

Req. ID	Security Requirement	Control
M1.SRP2a.1	Make passenger info unreadable to attackers during transmission between client and server	Encryption algorithm
M1.SRP2a.2	Make boarding pass unreadable to attackers during transmission between client and server	
M1.SRP5c.1	Make passenger info invisible before storing in data store	Encryption algorithm, monitor data access
M1.SRP4a.1	Filter abnormal requests at the server	Firewall, DoS Defence System
M1.SRP2b.1	Verify integrity of passenger info after transmission	Checksum algorithm
M1.SRP2b.2	Verify integrity of boarding pass after transmission	
M1.SRP3a.1	Filter input after receiving passenger check-in request	Filter input for special characters and keywords, use whitelist of acceptable inputs
M1.SRP5a.1	Monitor the data store at Server for malicious changes	Monitor and verify database signature changes
M1.SRP6.1	Filter confidential information from error messages and standard responses	Disable debug messages, use default error messages or error pages

5.3 Threats to Validity

Our validation method contains some degree of subjectivity. Throughout the validation process we only asked the opinion of one expert. Although we trust the feedback we received, opinions by nature are subjective and it would be good to collect opinions from other experts too.

Another limitation of our validation method is that we applied the security patterns only to five business processes. Although the processes are based on real life scenarios, they do not represent a sample of all the existing business processes. It would be interesting to apply the security patterns to business processes from other industries besides aviation, to see how well they conform in a different environment.

What should also be considered is that the security patterns were applied to the example business processes by the author of the patterns. Different people may have different observations of the security patterns applicability. It would be interesting to also get feedback from someone unfamiliar with the SRPs who has applied them to a business process.

The business processes we performed our validation on were created by the author of the security risk-oriented patterns. Although we do not consider this a real threat, as the business processes describe life scenarios and were also validated by an expert, it is still worth noting.

5.4 Discussion

5.4.1 Expert's Feedback

The expert's feedback to the secured business processes was approving. Security requirements suggested by the security patterns are relevant for reducing risks. The expert expressed confidence that our revised library of security risk-oriented patterns could be taken as a base for future development of a security patterns catalogue.

5.4.2 Observations

We made the following observations after application of the security risk-oriented patterns:

- The following SRPs could not be applied in any of the five business processes:
  SRP4b - Secure business services against resource exhaustion attacks. The one common resource in the five business processes is the data store. If an attacker performs many database connection pool entries at the same time, it is possible that it could affect the availability of a business service. However, we already apply SRP4a in all of the business processes which filters abnormal requests to the server. Thus in these five business processes SPR4b is not needed. We are still positive however, that SRP4b could be applicable in some business processes where users have more control over the system resources.
  SRP7 - Protect business services from deadlock attacks. We could not pinpoint anywhere in any of the business processes where this attack could be possible. Therefore, we cannot confirm this pattern's applicability. It could potentially be applied in a more complex business processes where there are many competing processes dependent on one-another.
  SRP8a - Secure system from brute force password attacks.
  SRP8b - Secure system from unauthorized access through common or default usernames and passwords.
  SRP8c - Secure the system from account lockout attacks. Unfortunately, none of the five business processes used passwords as a method of authentication. Therefore, we could not apply patterns SRP8a, SRP8b and SRP8b anywhere. We are still confident however, that these patterns are applicable in business processes that implement password authentication.
- The sequence of the security requirements can be different in business process models than it is in reality. When arranging the sequence of the security requirements in the business process models, we relied on a logical approach. For example, in the fuel service form issuing process (see Figure 78) we described, that the server verifies the integrity of fuel quantity info before proceeding with making the fuel quantity info readable. However, in reality it could be that the implementation chosen to satisfy these requirements performs message encryption and authentication in a reverse order (Krawczyk, 2001). There may be other security controls that can alter the sequence of security requirements as well. It is important that the sequence of the security requirements in a business process does not limit the choice between security controls. Thus it is necessary to explain to implementers of the business process, that the sequence of the security requirements depicted on a business process model may not always represent the end result.

6 Conclusions

This chapter summarises the answers to our research questions, discusses the limitations of the work and proposes objectives for future research.

6.1 Answers to Research Questions

In this thesis we were looking for answers to four research questions. The results of the thesis enable us to now answer these questions.

RQ: How can the library of security risk-oriented patterns be expanded to support development of secure distributed systems? This was the primary research question of our study. We explored two approaches on how the library of security risk-oriented patterns could be expanded. The goal of the first approach was to provide security risk-oriented patterns that would target security threats individually. The goal of the second approach was to produce additional security risk-oriented patterns to provide mitigations for security threats that were currently not addressed.

RQ1: Which security threats do the security risk-oriented patterns correspond to? We aligned the existing security risk-oriented patterns against the threat patterns taxonomy by Uzunov & Fernandez (2013). From this we observed, that the five original security risk-oriented patterns address altogether ten security threats. These security threats are the following: *(i)* invoking unauthorized operations, *(ii) m*essage secrecy violation, *(iii)* message integrity violation, *(iv)* injection, *(v)* process overflow attack, *(vi)* message flooding, *(vii)* resource exhaustion, *(viii)* corruption, *(ix)* track erasing, and *(x)* unauthorized access. In this thesis we propose new security risk-oriented patterns that address each of these security threats individually. These security patterns support more effective engineering of security in distributed systems, because they enable business analysts to explicitly mitigate risks that are actual in a specific business process.

RQ2: Which additional security risks and countermeasures should be considered to be able to apply security risk-oriented patterns for securing business processes in distributed systems? We identified five additional security threats that should be considered when securing business processes in distributed systems. These threats are the following: *(i)* output information disclosure, *(ii)* targeted process crashing, *(iii)* password attacks (guessing, brute force, rainbow tables, etc.), *(iv)* use of default credentials, and *(v)* account lockout attack. Based on these security threats, we created five security risk-oriented patterns that provide countermeasures for each of these threats.

RQ3: What is the applicability of the security risk-oriented patterns in distributed systems? We examined the applicability of the security risk-oriented patterns in five business processes from the aviation industry. The business processes were developed based on a turnaround process described by (Nõukas, 2015). We applied the security risk-oriented patterns to the business processes using the SREBP methodology. The business processes were then enhanced with security requirements derived from the security patterns and presented using the security risk-aware BPMN modelling language. We submitted the secured business processes for review to an expert from Tallinn University of Technology who has experience with business processes used in the airline industry. Our validation efforts confirmed the applicability of eleven security risk-oriented patterns out of sixteen.

6.2 Limitations

As we learned from the validation chapter, we could not confirm the applicability of five security risk-oriented patterns. We are positive however, that this does not necessarily mean

they are inapplicable in business processes. Rather, the example business processes did not expose those specific risks.

6.3 Future work

Although throughout this thesis we managed to expand the library of security risk-oriented patterns to support development of secure distributed systems, the library still does not address all the security threats out there. As we already discussed in chapter 3, there are other resources available for threat patterns besides Uzunov & Fernandez's threat pattern library (2013). Some best-known examples are Mitre's CAPEC and Microsoft's STRIDE. We would like to investigate both of these repositories in order to expand the security patterns library even further.

We would also like to test the applicability of the security risk-oriented patterns in other environments besides the aviation industry in order to see how well they perform. This could potentially provide us with verification of the applicability of the security patterns that were not applied in the airline business processes.

7 References

Ahmed, N., & Matulevičius, R. (2011). A template of security risk patterns for business processes. In Perspectives in Business Informatics Research (pp. 123-130). Riga: JUMI Publishing House.

Ahmed, N. (2014). Deriving Security Requirements from Business Process Models. PhD Thesis, University of Tartu, Tartu.

Ahmed, N., & Matulevičius, R. (2014). Securing business processes using security risk-oriented patterns. Computer Standards & Interfaces. 36(4), 723-733.

Alberts C. J., & Dorofee A. J. (2001). OCTAVE Method Implementation Guide Version 2.0. Pennsylvania: Carnegie Mellon University, Software Engineering Institute.

Altuhhova, O., Matulevičius, R., & Ahmed, N. (2013). An Extension of Business Process Model and Notation for Security Risk Management. International Journal of Information System Modeling and Design, 4(4), 93-113.

Andress, J. (2011). The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice. USA: Syngress.

Barbará, D., Goel, R., & Jajodia, S. (2000). Using Checksums to Detect Data Corruption. Advances in Database Technology - EDBT 2000, 1777, 136-149.

Behnia, A., Rashid, R. A., & Chaudhry, J. A. (2012). A Survey of Information Security Risk Analysis Methods. Smart Computing Review, 2(1), 79-94.

Boehm, B. (1989). Software Risk Management. In ESEC '89 Proceedings of the 2nd European Software Engineering Conference.

Braber, F., Hogganvik, I., Lund, M. S., Stølen, K., & Vraalsen, F. (2007). Model-based Security Analysis in Seven Steps — a Guided Tour to the CORAS Method. BT Technology Journal, 25(1), 101–117.

Bushcmann, F., Henney, K., & Schmidt, D. C. (2007). Pattern Oriented Software Architecture: On Patterns and Pattern Languages, Volume 5. Chichester, England: Wiley.

Buyens, K., Gregoire, J., De Win, B., Scandariato, R., Joosen, W. (2007). Similarities and differences between CLASP, SDL, and Touchpoints: the activity-matrix (CW Reports vol:CW501). University of Leuven, Department of Computer Science.

Chowdhury, M. J. M., Matulevičius, R., Sindre, G., & Karpati, P. (2012). Aligning Mal-activity Diagrams and Security Risk Management for Security Requirements Definitions. In B. Regnell, D. Damian (Eds.), Requirements Engineering: Foundation for Software Quality (pp. 132-139). Essen: Springer-Verlag Berlin Heidelberg.

DCSSI. (2004). EBIOS – Expression of Needs and Identification of Security Objectives. French Network and Information Security Agency.

Dijkman, R. M., Dumas, M., & Ouyang, C. (2007). Formal Semantics and Analysis of BPMN Process Models using Petri Nets. Information and Software Technology, 50(12), 1281-1294.

Dubois, É., Heymans, P., Mayer, N., & Matulevičius, R. (2010). A Systematic Approach to Define the Domain of Information System Security Risk Management. In S. Nurcan, C. Salinesi, C. Souveyet, J. Ralyté (Eds.), Intentional Perspectives on Information Systems Engineering (pp. 289-306): Springer Berlin Heidelberg.

Firesmith, D. (2007). Engineering Safety and Security Related Requirements for Software Intensive Systems (Full Day Tutorial). Carnegie Mellon University, Software Engineering Institute.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. USA: Addison-Wesley.

Howard, M., & Lipner, S. (2006). The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software. Redmond, Washington: Microsoft Press.

Insight Consulting. (2003). CRAMM (CCTA Risk Analysis and Management Method) User Guide version 5.0. SIEMENS.

Jürjens, J. (2005). Secure Systems Development with UML. Berlin: Springer-Verlag Berlin Heidelberg.

Krawczyk, H. (2001). The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In J. Kilian (Ed.), Advances in Cryptology: Vol. 2139. CRYPTO 2001 (pp. 310 – 331): Springer Berlin Heidelberg.

Lund, M. S., Solhaug, B., & Stølen, K. (2011). Model-Driven Risk Analysis: The CORAS Approach. Berlin: Springer-Verlag Berlin Heidelberg.

Mayer, N. (2009). Model-based Management of Information System Security Risk. University of Namur, Computer Science.

Matulevičius, R. (2012). Comparing Modelling Languages for Information Systems Security Risk Management. N. Seyff, & A. Koziolek (Eds.), Modelling and Quality in Requirements Engineering (197 - 210): Verlagshaus Monsenstein und Vannerdat.

Matulevičius, R., Mayer, N., & Heymans, P. (2008). Alignment of Misuse Cases with Security Risk Management. S. Jakoubi, S. Tjoa, & E. R. Weippl (Eds.), Third

International Conference on Availability, Reliability and Security (1397 - 1404): Barcelona.

N. Seyff, & A. Koziolek (Eds.), Modelling and Quality in Requirements Engineering (197 - 210): Verlagshaus Monsenstein und Vannerdat.

McGraw, G. (2006). Software Security: Building Security In. Westford, Massachusetts: Addison-Wesley Professional.

Mead, N. R., Hough, E. D., & Stehney II, T. R. (2005). Security Quality Requirements Engineering (SQUARE) Methodology (Tech. Rep.). Pittsburgh, PA: Carnegie Mellon University, Software Engineering Institute.

Mellado, D., Fernandez-Medina, E., & Piattini, M. (2007). Common criteria based security requirements engineering process for the development of secure information systems. Computer Standards & Interfaces, 29(2), 244-253.

Mouratidis, H., & Giorgini, P. (2007). Secure Tropos: A Security-Oriented Extension of the Tropos Methodology. International Journal of Software Engineering and Knowledge Engineering, 17(2), 285-309.

Nõukas, R. (2015). Service Brokering Environment for an Airline. Master's thesis, Tallinn University of Technology, Tallinn.

Schumacher, M. (2003). Security Engineering with Patterns: Origins, Theoretical Models and New Applications. Germany: Springer.

Schumacher, M., Fernandez, E., Hybertson, D., & Buschmann, F. (2005). Security Patterns: Integrating Security and Systems Engineering. Germany: John Wiley & Sons.

Secure Software, Inc. (2005). The CLASP Application Security Process. Retrieved November 10, 2015, from https://www.ida.liu.se/~TDDC90/literature/papers/clasp_external.pdf

Sindre, G. (2007). Mal-Activity Diagrams for Capturing Attacks on Business Processes. In P. Sawyer, B. Paech, & P. Heymans (Eds.), Requirements Engineering: Foundation for Software Quality: Vol 4542. Lecture Notes in Computer Science (pp. 355-366). Berlin: Springer Berlin Heidelberg.

Sindre, G., & Opdahl, A. L. (2001). Capturing Security Requirements through Misuse Cases. Retrieved November 10, 2015, from http://folk.uio.no/nik/2001/21-sindre.pdf

Sindre, G., & Opdahl, A. L. (2005). Eliciting Security Requirements with Misuse Cases. Requirements Engineering, 10(1), 34-44.

Silver, B. (2009). BPMN Method and Style: A Levels-based Methodology for BPMN Process Modeling and Improvement using BPMN 2.0. Cody-Cassidy Press.

Soomro, I., & Ahmed, N. (2012). Towards Security Risk-oriented Misuse Cases. In M. La Rosa & P. Soffer (Eds.), Lecture Notes in Business Information Processing: Vol. 132. Business Process Management Workshops (pp. 689-700). Berlin: Springer Heidelberg.

Uzunov, A. V., & Fernandez, E. B. (2013). An extensible pattern-based library and taxonomy of security threats for distributed systems. Computer Standards & Interfaces, 36(4), 734-747.

Appendix

The appendix includes all the additional material developed throughout this thesis. It is divided into four sections. They are the following:

I. **BPMN Models of Security Risk-oriented Patterns.** This appendix contains all the asset, risk and risk-treatment models developed during the revision of security risk-oriented patterns (see chapter 4.1).
II. **Conformation of Security Risk-oriented Patterns and Uzunov and Fernandez's Threat Patterns.** This appendix contains the tabular alignment of all the security risk-oriented patterns to (Uzunov & Fernandez, 2013) threat patterns (see chapters 4.1 and 4.2).
III. **Application of Security Risk-oriented Patterns to Five Example Business Processes.** This appendix includes the additional risk and risk-treatment models developed during the validation process (see chapter 5.1). It also contains the descriptions of the four other aviation business processes used for validation.
IV. **License.** This is the non-exclusive licence that permits the reproduction and publication of this thesis.

I. BPMN Models of Security Risk-oriented Patterns

SRP1a

This pattern secures data from unauthorized access.

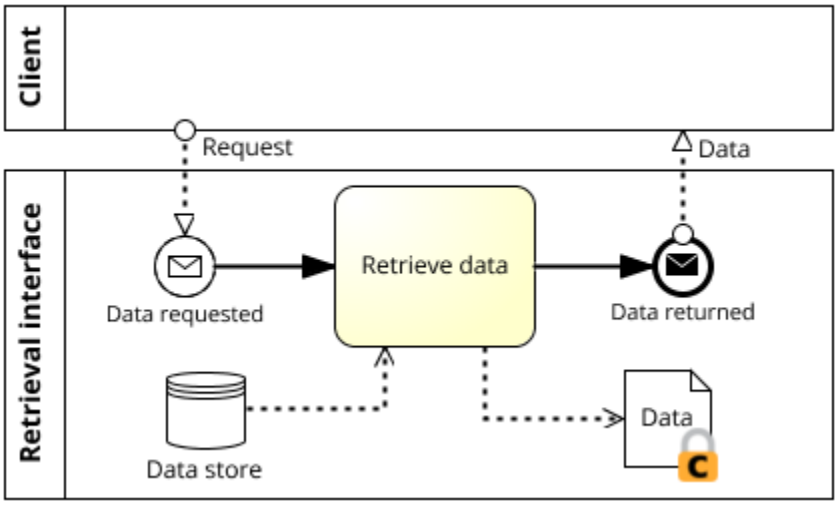


Figure 27. SRP1a: Business Process Model with Security Objectives

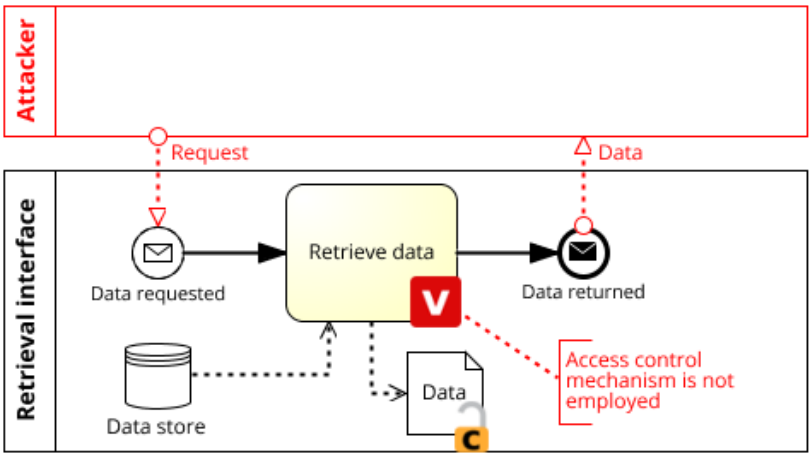


Figure 28. SRP1a: Security Risk-oriented Business Process Model

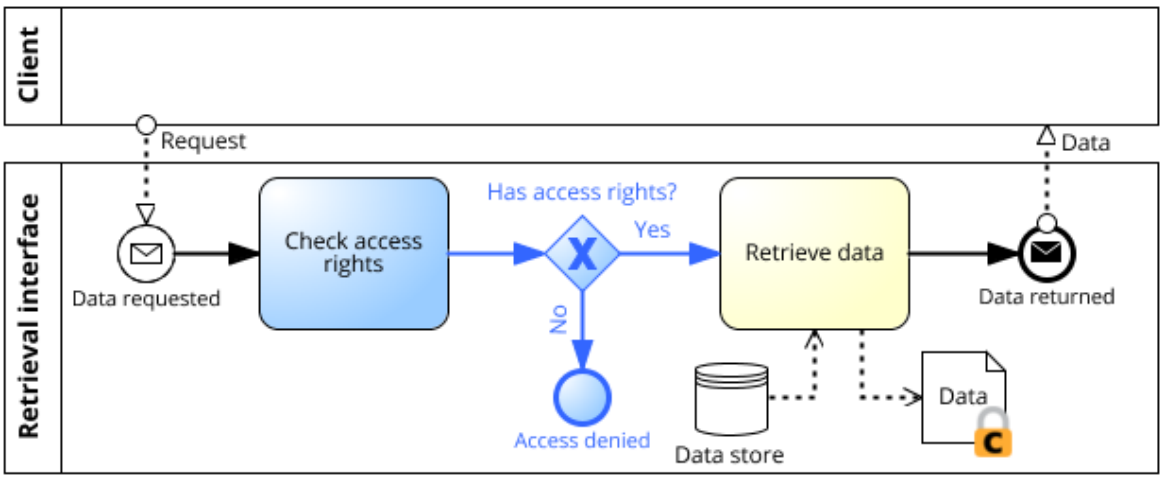


Figure 29. SRP1a: Security Requirements for Business Process

SRP1b

This pattern protects business services from unauthorized access.

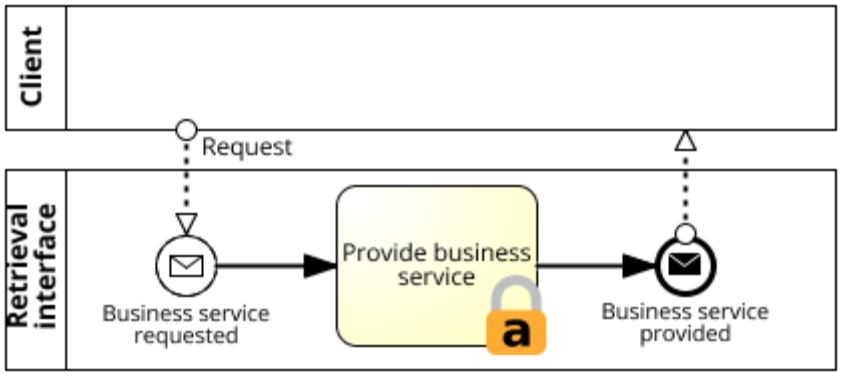


Figure 30. SRP1b: Business Process Model with Security Objectives

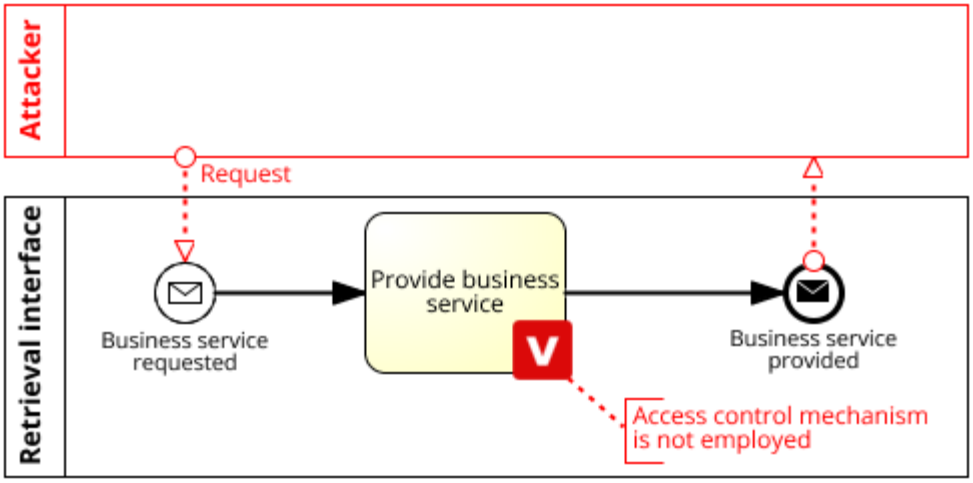


Figure 31. SRP1b: Security Risk-oriented Business Process Model

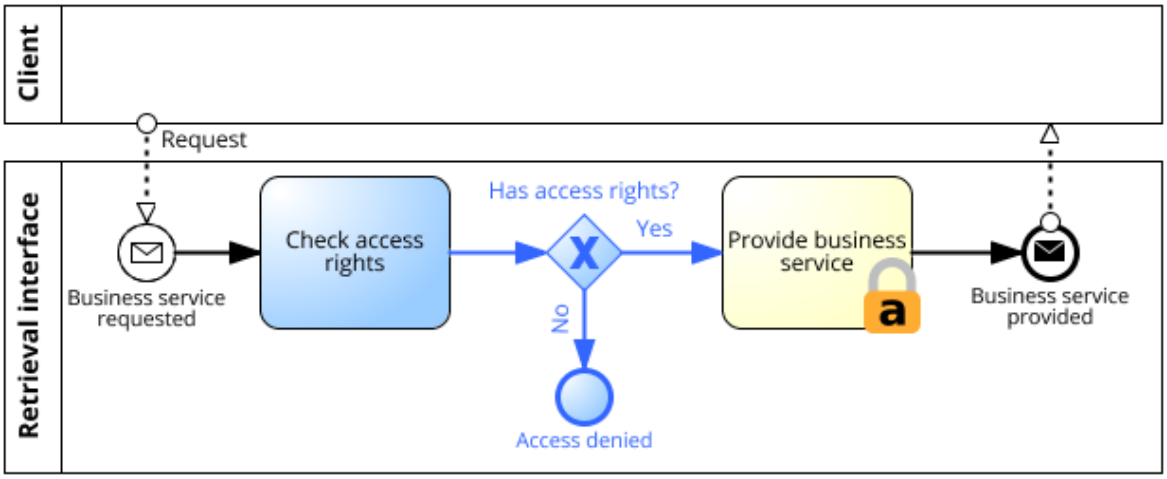


Figure 32. SRP1b: Security Requirements for Business Process

SRP2a

This pattern helps ensure data confidentiality during transmission between business entities.

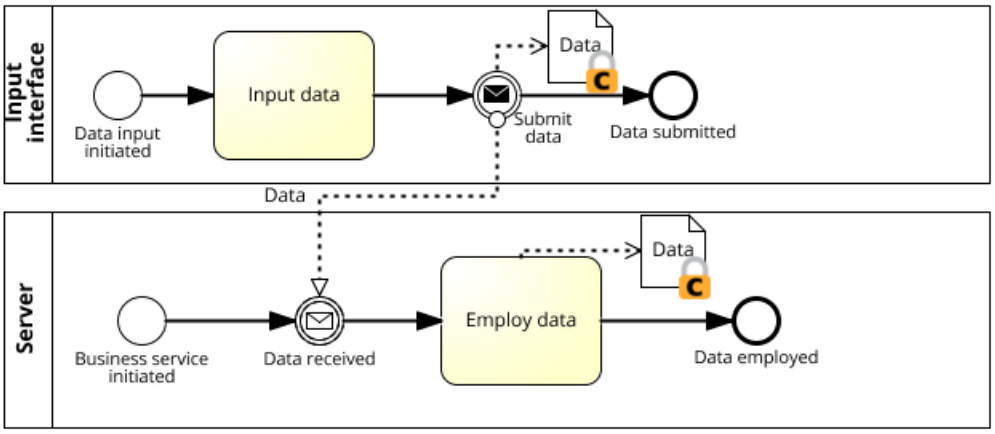


Figure 33. SRP2a: Business Process Model with Security Objectives

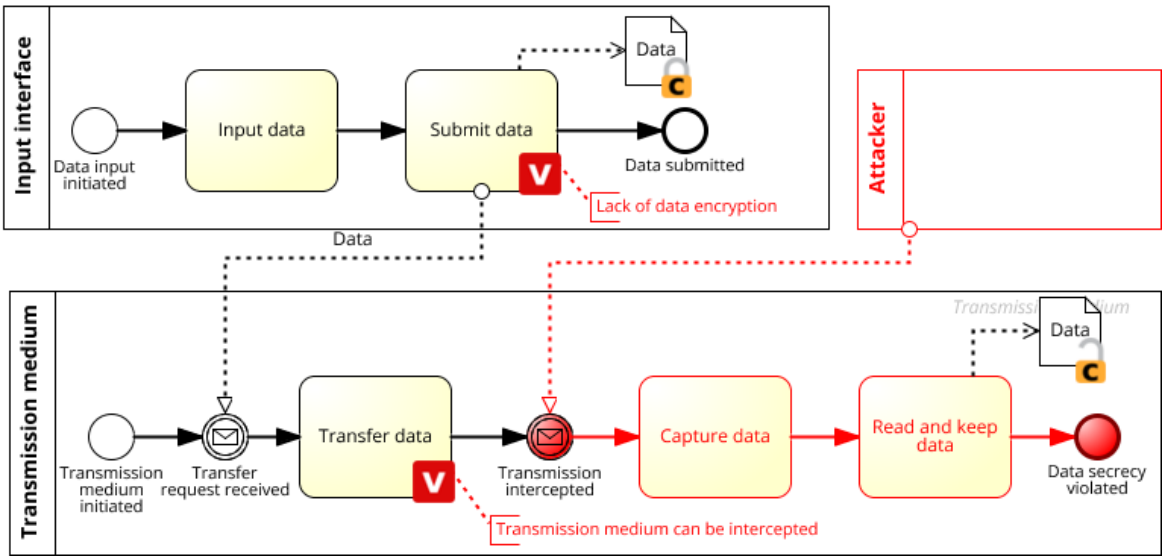


Figure 34. SRP2a: Security Risk-oriented Business Process Model

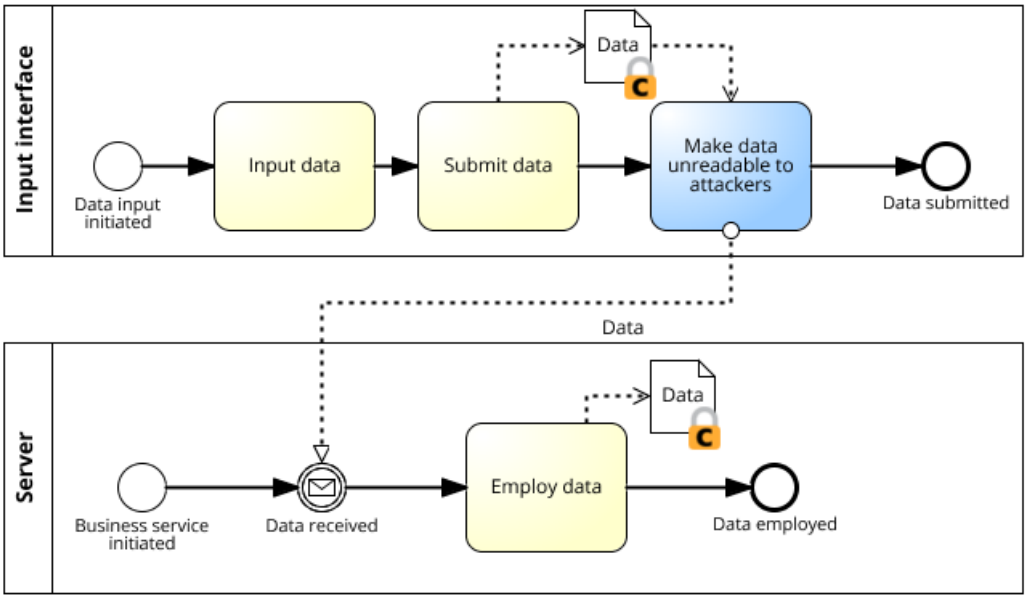


Figure 35. SRP2a: Security Requirements for Business Process

SRP2b

This pattern helps ensure data integrity during transmission between business entities.

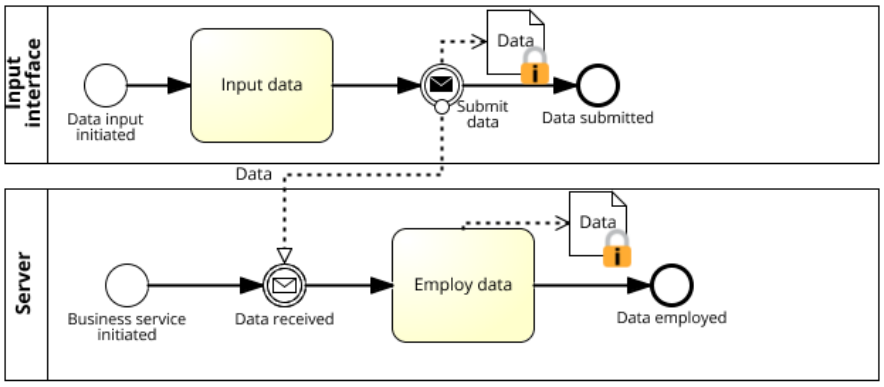


Figure 36. SRP2b: Business Process Model with Security Objectives

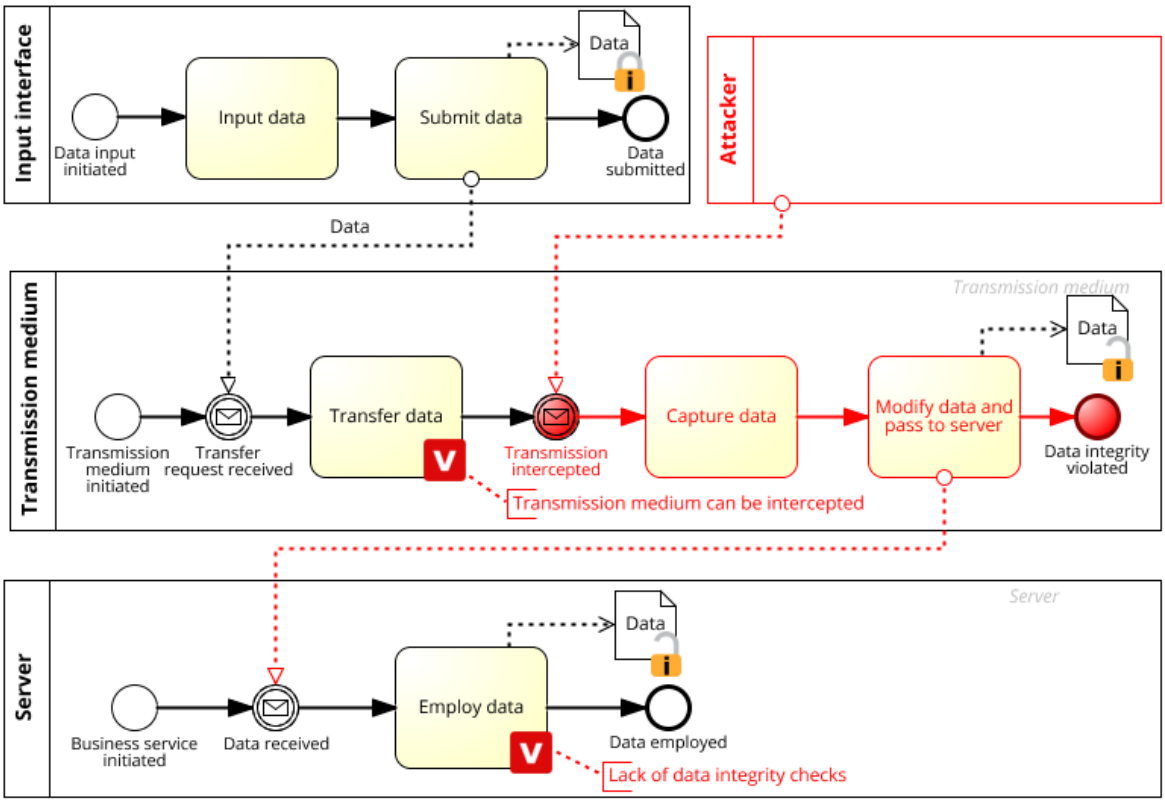


Figure 37. SRP2b: Security Risk-oriented Business Process Model

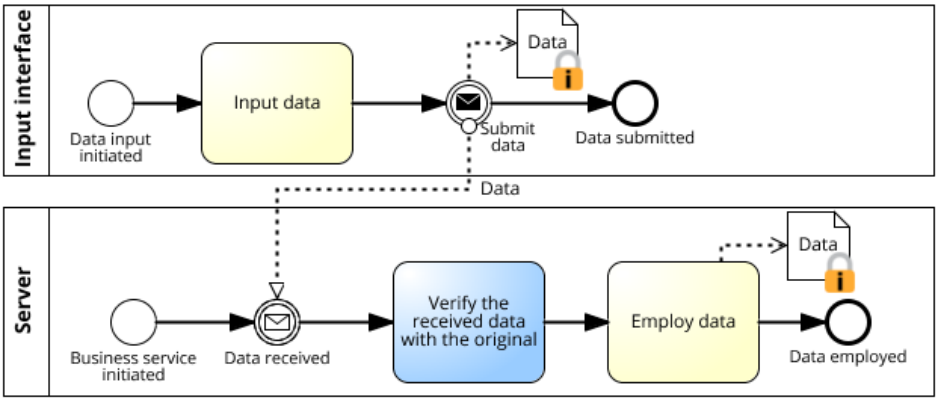


Figure 38. SRP2b: Security Requirements for Business Process

SRP3a

This pattern secures business activities from injection of malicious scripts.

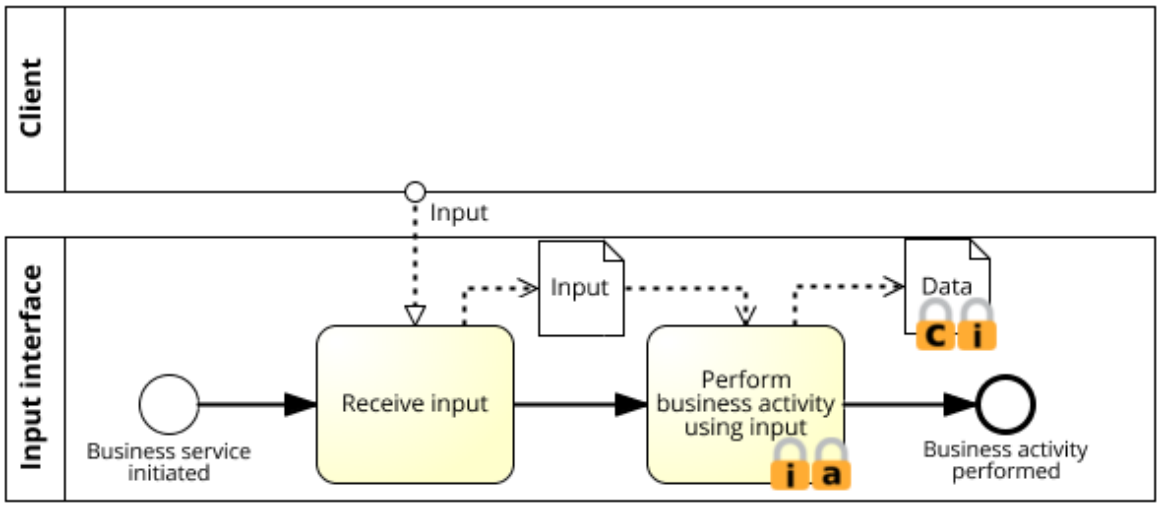


Figure 39. SRP3a: Business Process Model with Security Objectives

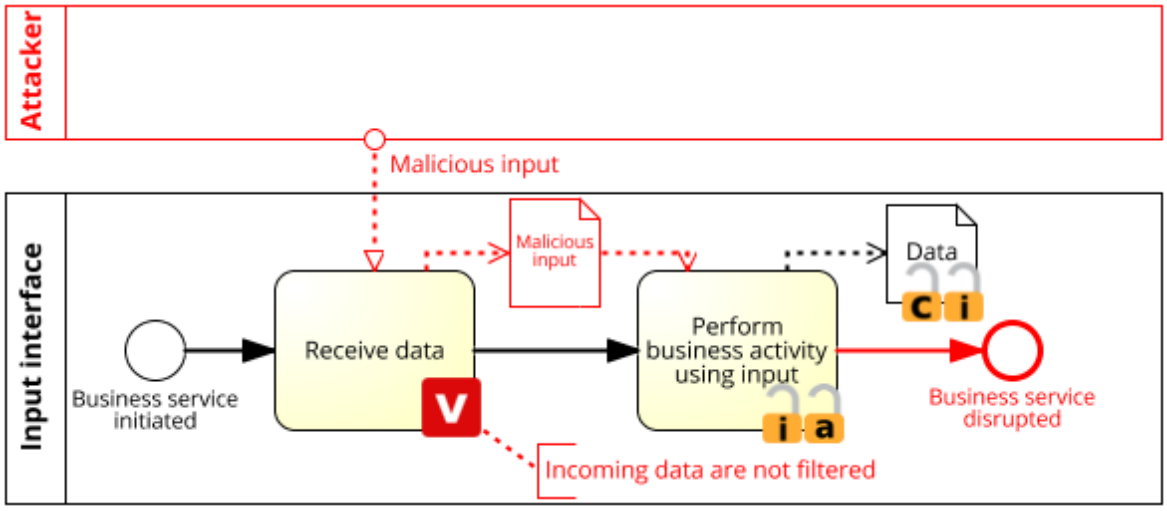


Figure 40. SRP3a: Security Risk-oriented Business Process Model

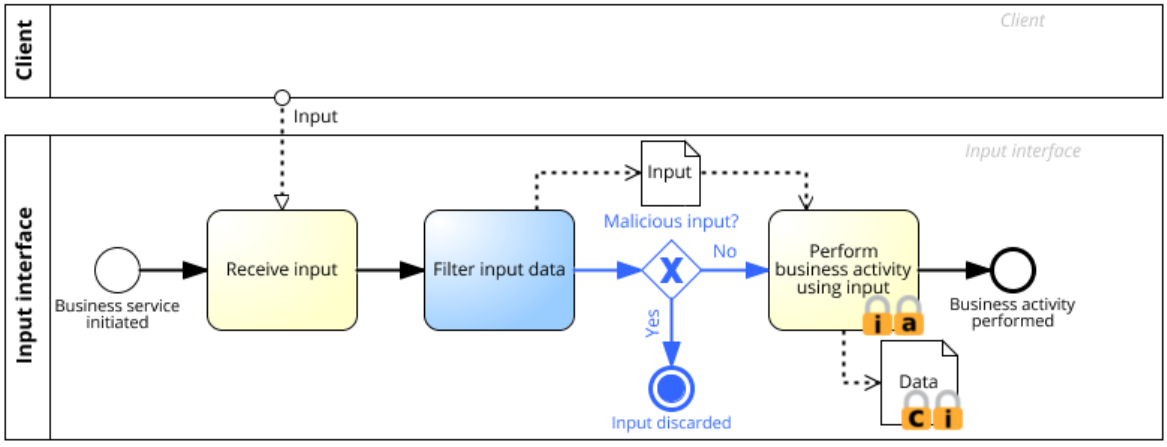


Figure 41. SRP3a: Security Requirements for Business Process

SRP3b

This pattern secures business activities from buffer overflow attacks. The security objectives and requirements in this pattern are identical to SRP3a.

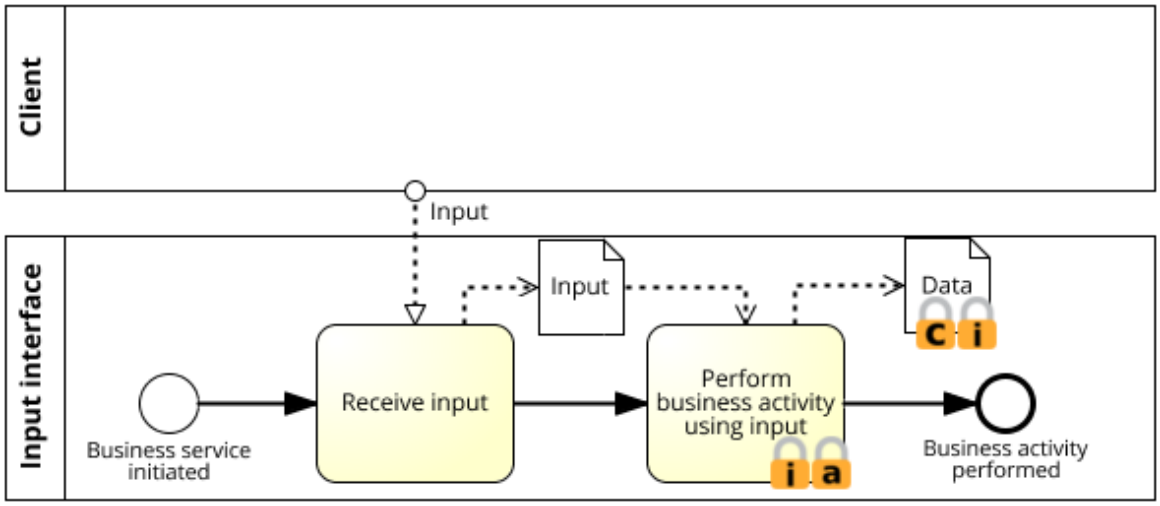


Figure 42. SRP3a and SRP3b: Business Process Model with Security Objectives

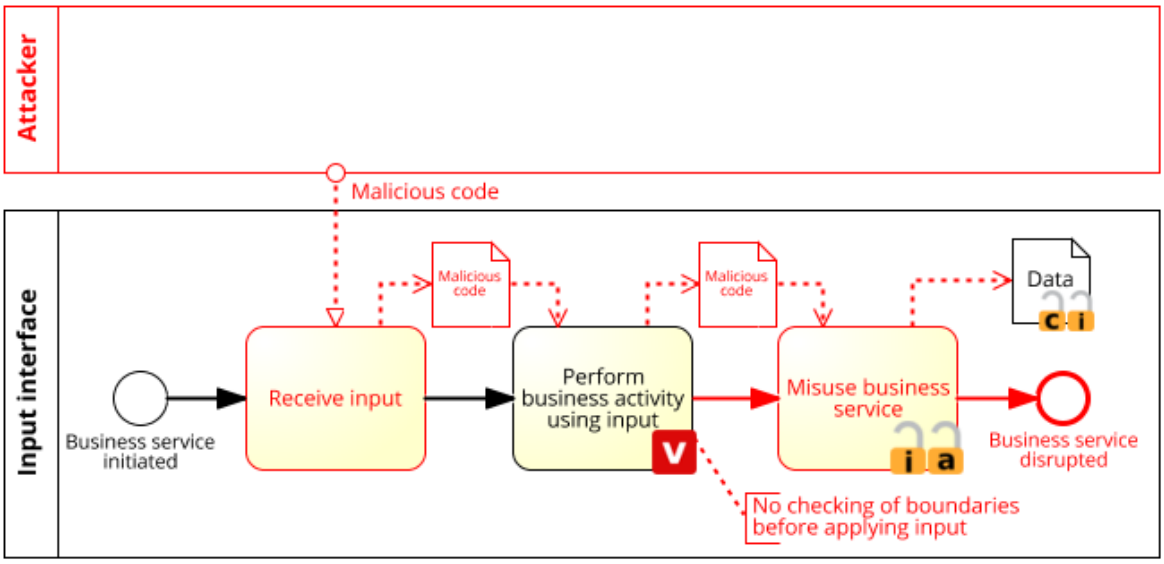


Figure 43. SRP3b: Security Risk-oriented Business Process Model

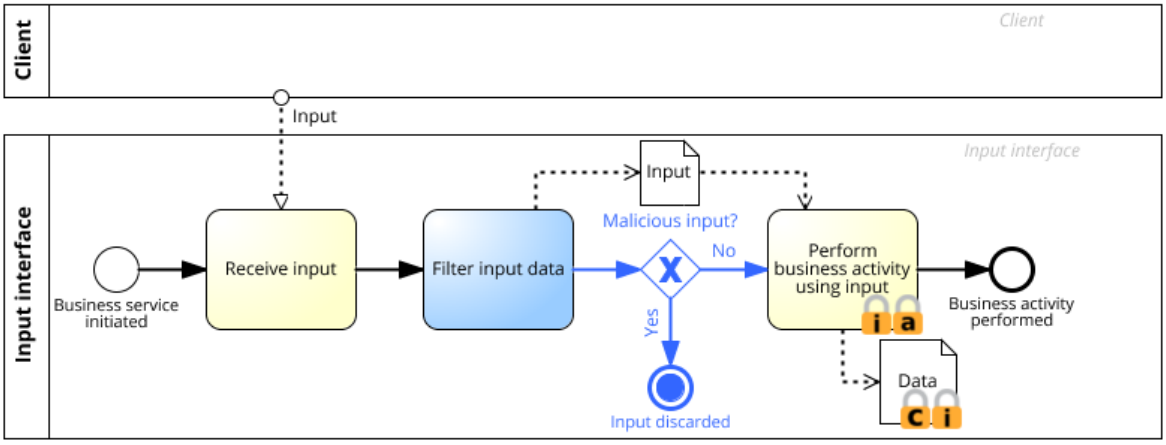


Figure 44. SRP3b: Security Requirements for Business Process

SRP4a

This pattern secures business services against message flooding attacks.

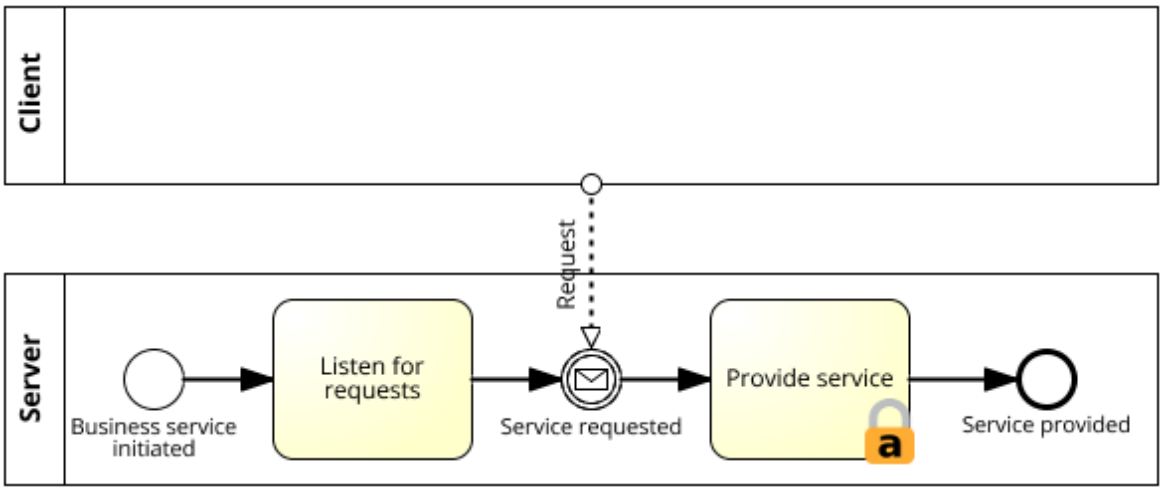


Figure 45. SRP4a: Business Process Model with Security Objectives

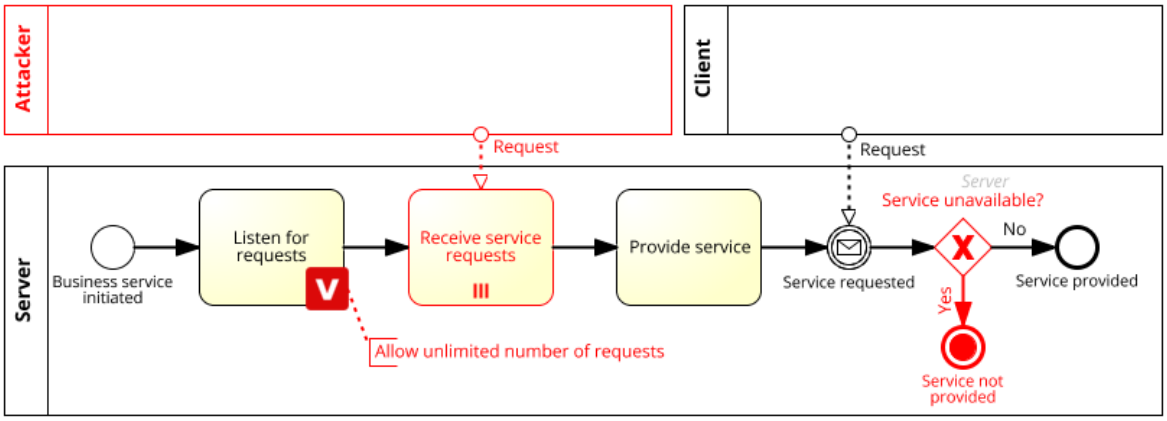


Figure 46. SRP4a: Security Risk-oriented Business Process Model

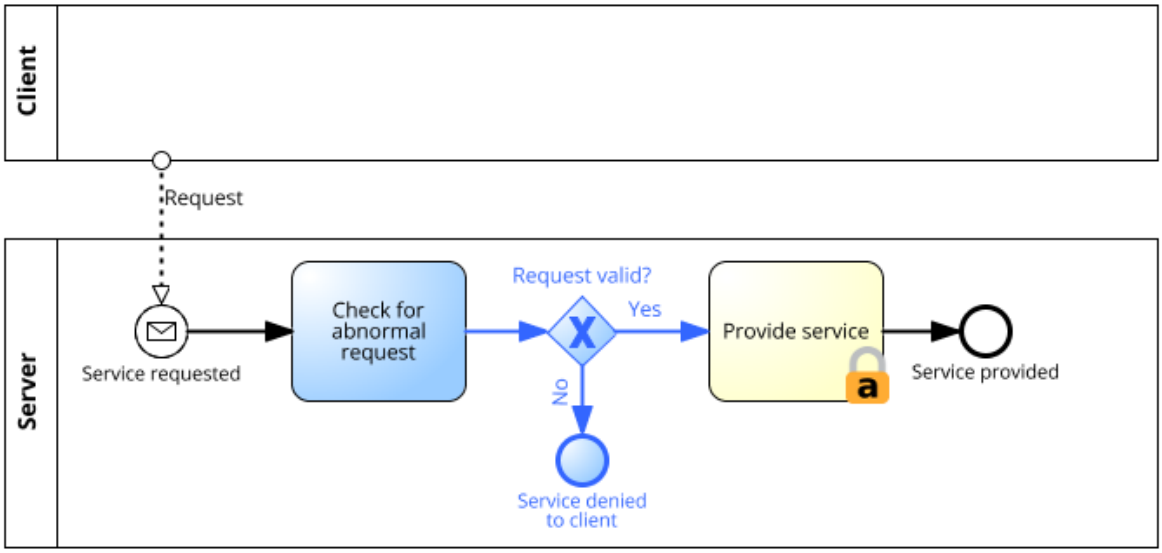


Figure 47. SRP4a: Security Requirements for Business Process

SRP4b

This pattern secures business services against resource exhaustion attacks. The security objectives of this pattern are identical with SRP4a.

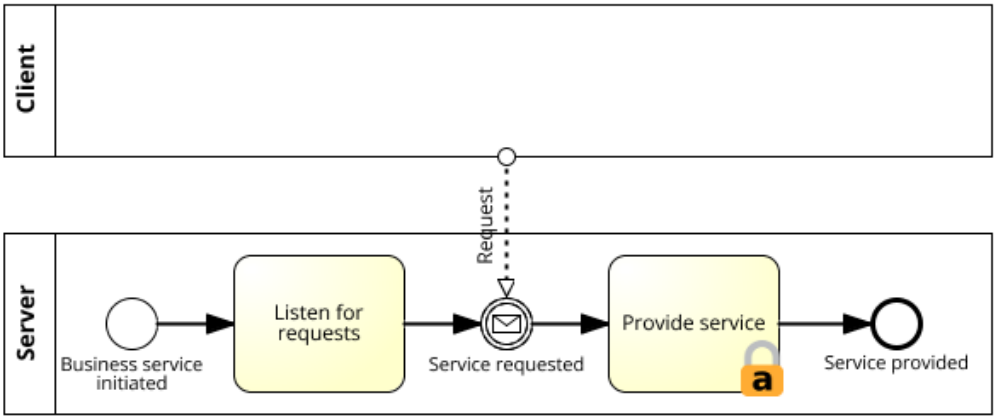


Figure 48. SRP4b: Business Process Model with Security Objectives

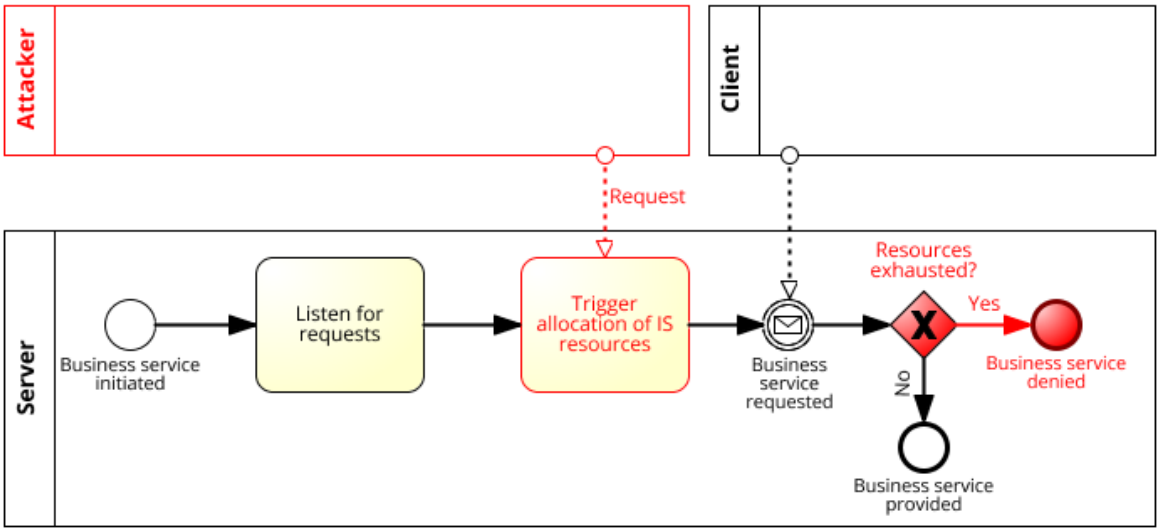


Figure 49. SRP4b: Security Risk-oriented Business Process Model

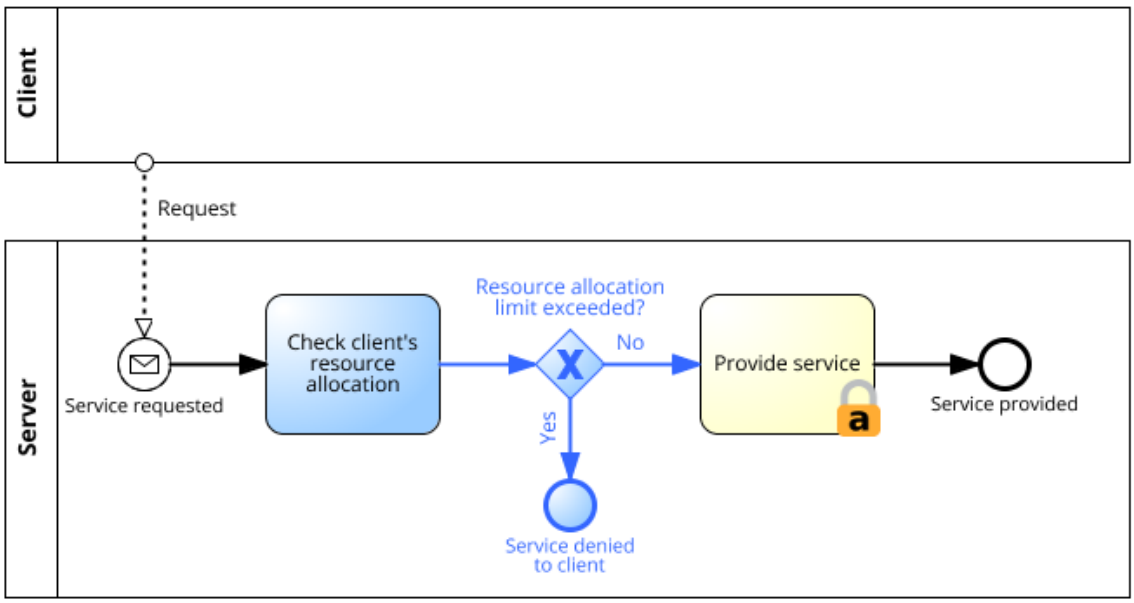


Figure 50. SRP4b: Security Requirements for Business Process

SRP5a

This pattern helps secure data in data store from corruption.

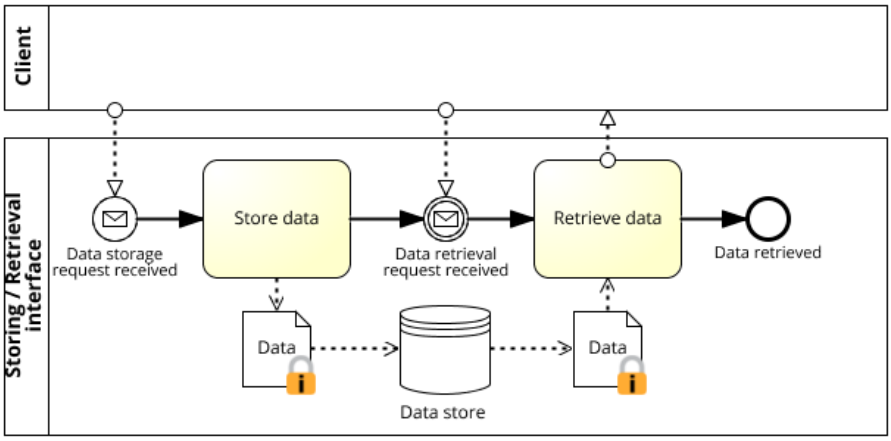


Figure 51. SRP5a: Business Process Model with Security Objectives

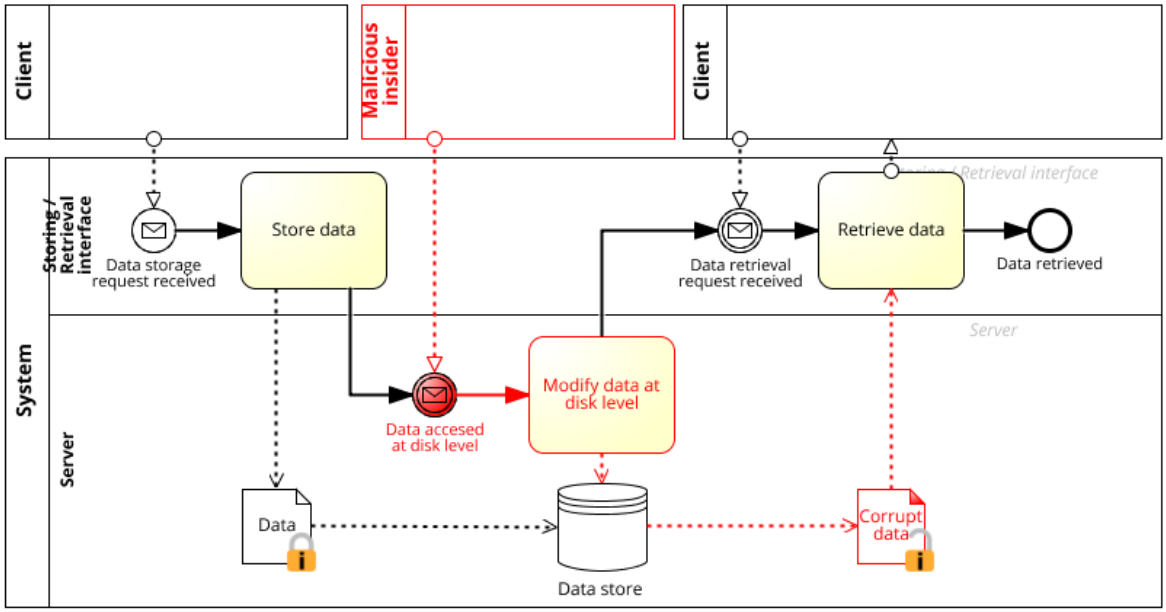


Figure 52. SRP5a: Security Risk-oriented Business Process Model

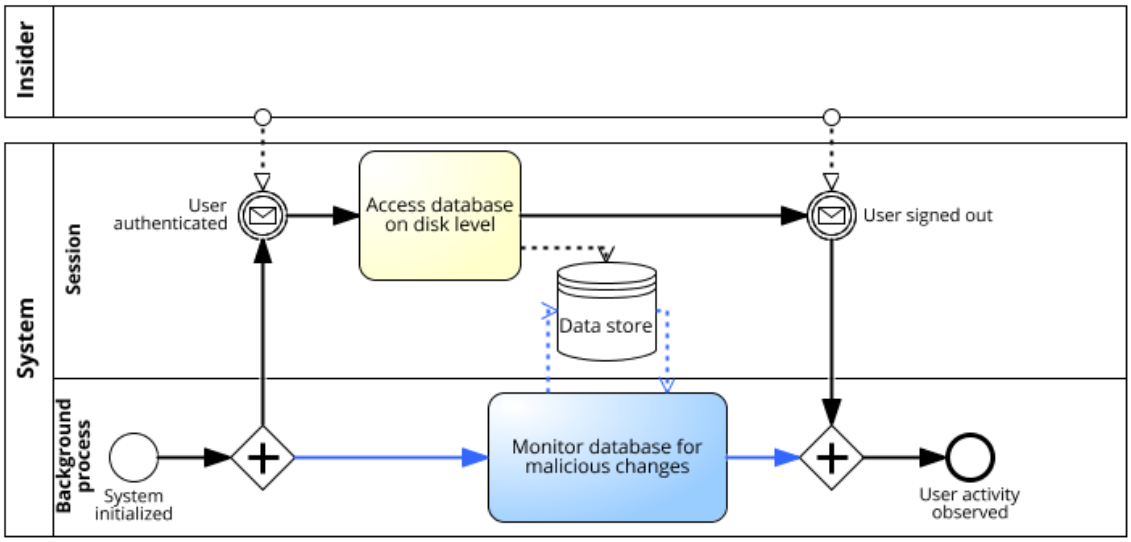


Figure 53. SRP5a: Security Requirements for Business Process

SRP5b

This pattern helps secure the integrity of auditing information. The security objectives and requirements for this pattern are identical to SRP5a.

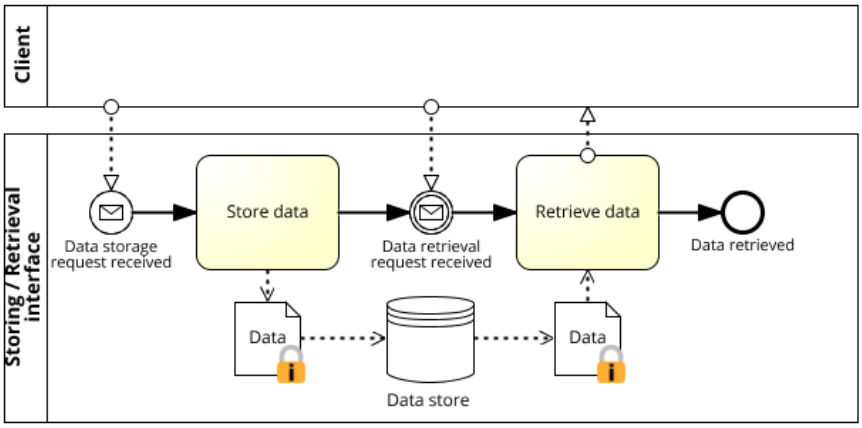


Figure 54. SRP5b: Business Process Model with Security Objectives

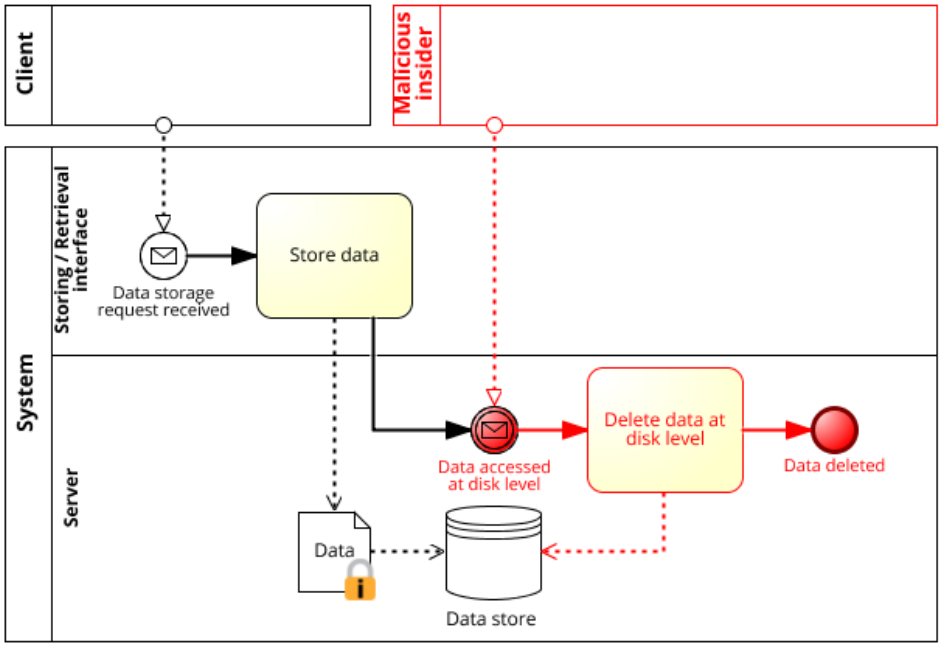


Figure 55. SRP5b: Security Risk-oriented Business Process Model

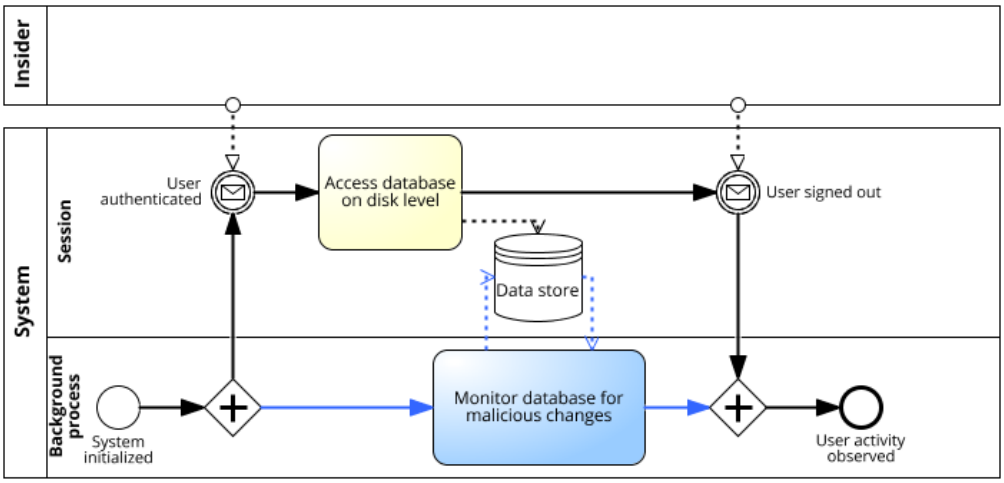


Figure 56. SRP5b: Security Requirements for Business Process

SRP5c

This pattern secures data in data store from unauthorized access by malicious insiders.

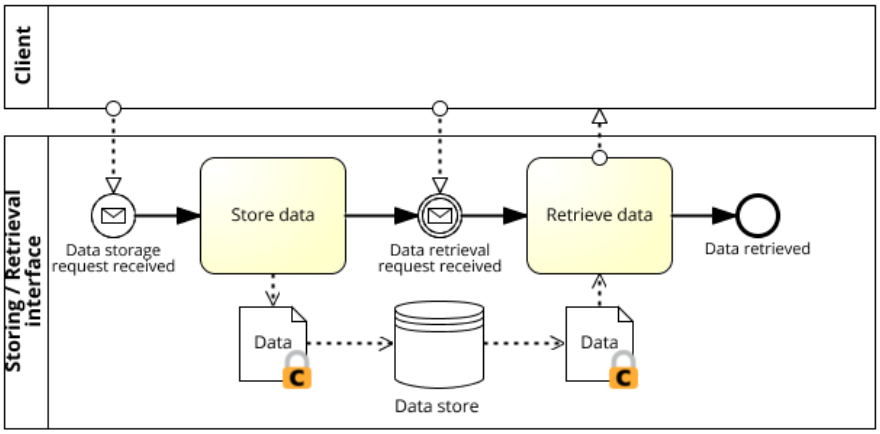


Figure 57. SRP5c: Business Process Model with Security Objectives

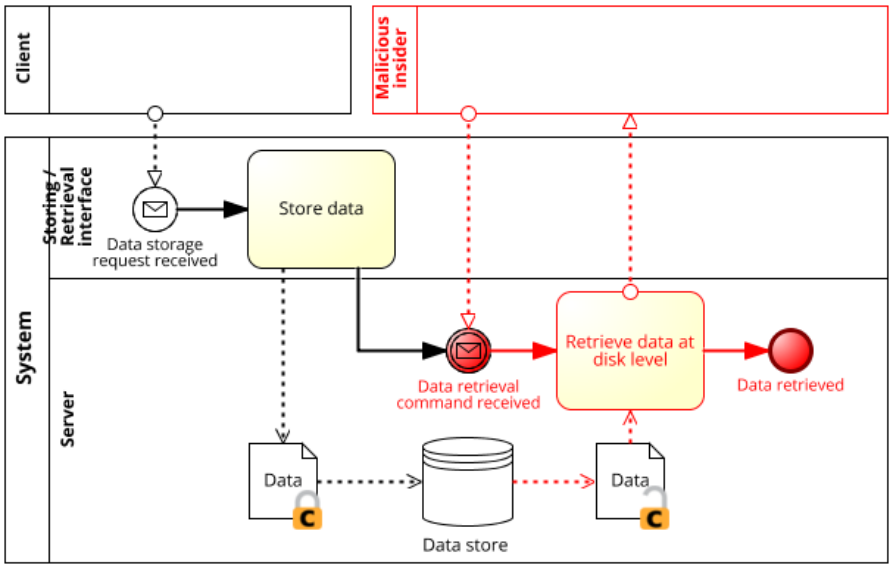


Figure 58. SRP5c: Security Risk-oriented Business Process Model

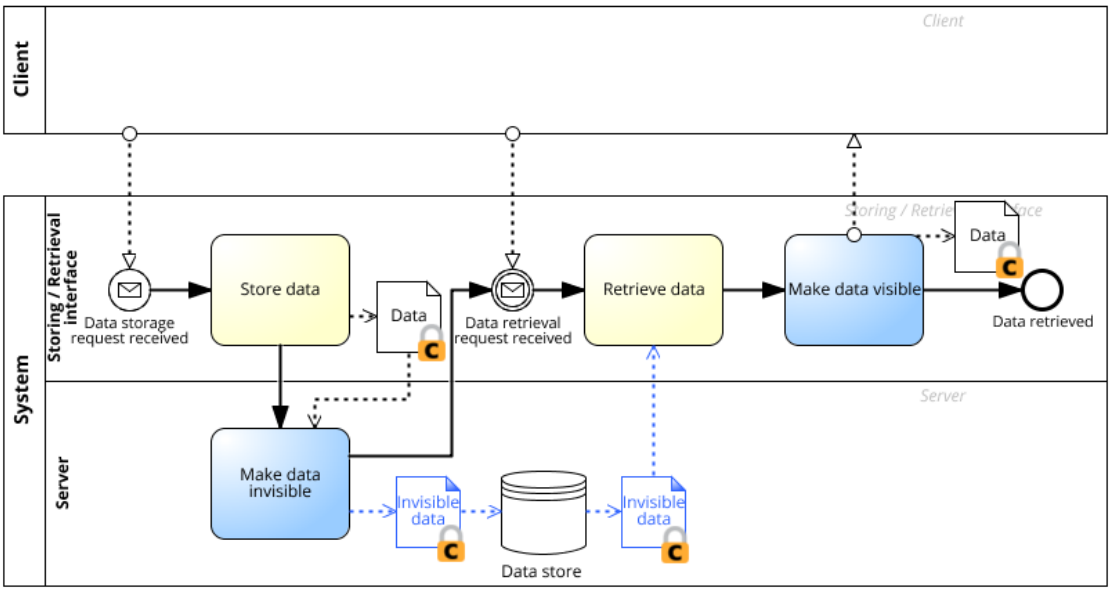


Figure 59. SRP5c: Security Requirements for Business Process

II. Conformation of Security Risk-oriented Patterns and Uzunov and Fernandez's Threat Patterns

Table 19. Conformation of SRP1a and UFP 8.2, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP1a. Secure data from unauthorized access	UFP 8.2 Invoking unauthorized operations (Uzunov & Fernandez, 2013, Table 9)
Business asset	Data stored in server and requested by user	Operations
IS asset	Input interface, transmission channel, server	User interaction, e.g., input ports Distribution control Resource management
Security criteria	Data confidentiality	
Impact	Loss of data confidentiality, compromise of business	
Threat	An attacker gets hold of confidential data	A software component invokes operations on another component without being authorized to do so.
Threat agent	An attacker acting as a client with the intention of getting a hold of confidential data on the server	
Attack method	Request confidential data from the server by appearing as a client. Exploit data: read and save for personal use	
Vulnerability	Lack of access permission checks at the server	
Risk treatment	Risk reduction	
Security requirement	Perform access permission checks for data retrieval by clients	Authorization
Control	Define user levels with appropriate access authorization with credentials, monitor data access	

Table 20. Conformation of SRP1b and UFP 8.2, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	**Threat pattern for distributed systems**
Pattern	SRP1b. Secure business service from unauthorized access	UFP 8.2 Invoking unauthorized operations (Uzunov & Fernandez, 2013, Table 9)
Business asset	Business service	Operations
IS asset	Input interface, transmission channel, server	User interaction, e.g., input ports Distribution control Resource management
Security criteria	Confidentiality of business process	
Impact		
Threat	An attacker gets access to business service without being authorized to do so	A software component invokes operations on another component without being authorized to do so.
Threat agent	An attacker acting as a client with the intention of using a business service	
Attack method	Request business service from the server by appearing as a client.	
Vulnerability	Lack of access permission checks at the server	
Risk treatment	Risk reduction	
Security require-ment	Perform access permission checks for business process access by clients	Authorization
Control	Define user levels with appropriate access authorization with credentials, monitor business service access	

Table 21. Conformation of SRP2a and UFP 2.1, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat patterns for distributed systems
Pattern	SRP2a. Secure data from secrecy violation during transmission between business entities	UFP 2.1 Message secrecy violation (Uzunov & Fernandez, 2013, Table 3)
Business asset	Data being transmitted	Messages
IS asset	Input interface, transmission channel, server	Communication, i.e., message channels, networking infrastructure
Security criteria	Confidentiality of data	Secrecy
Impact	Harm of business asset (data), loss of data confidentiality	
Threat	An attacker intercepts the transmission medium by acting as a proxy	Messages in transit are intercepted and their contents read by an attacker
Threat agent	An attacker with an intention of intercepting the transmission medium by acting as a proxy between the input interface and the server	
Attack method	Interception of transmission medium by installation of a proxy between input interface and server. Misuse of transmitted data: capture, read, store	
Vulnerability	Vulnerability of the transmission method to be intercepted Poor encryption algorithm, or lack of encryption being used at the input interface and the server	
Risk treatment	Risk reduction	
Security requirement	Make data unreadable to attacker	Secure communication
Control	Encryption algorithm	Message encryption

Table 22. Conformation of SRP2b and UFP 2.2, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat patterns for distributed systems
Pattern	SRP2b. Secure data from integrity violation during transmission between business entities	UFP 2.2 Message integrity violation (Uzunov & Fernandez, 2013, Table 3)
Business asset	Data being transmitted	Messages
IS asset	Input interface, transmission channel, server	Communication, i.e., message channels, networking infrastructure
Security criteria	Integrity of data	Integrity
Impact	Harm of business asset (data), loss of data integrity	
Threat	An attacker intercepts the transmission medium by acting as a proxy	Messages in transit are intercepted and modified, replaced, corrupted or simply deleted by an attacker
Threat agent	An attacker with an intention of intercepting the transmission medium by acting as a proxy between the input interface and the server	
Attack method	Interception of transmission medium by installation of a proxy between input interface and server. Misuse of transmitted data: capture, modify and pass to the server	
Vulnerability	Vulnerability of the transmission medium to be intercepted Poor or lack of message integrity validation at the server	
Risk treatment	Risk reduction	
Security requirement	Verify integrity of the received data	Secure communication
Control	Checksum algorithm	Message hashing, error detection codes

Table 23. Conformation of SRP3a and UFP 4.1, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP3a. Secure business activity from injection attacks	UFP 4.1 Injection (Uzunov & Fernandez, 2013, Table 5)
Business asset	Data being entered, business service	Messages
IS asset	Input interface, transmission channel, server	Communication, i.e., messages User interaction, i.e., input ports Data/storage management Distribution control
Security criteria	Integrity and confidentiality of data, integrity and availability of business service	
Impact	Harm of business asset (data), loss of data confidentiality and integrity, malicious scripts stored in server, loss of business service availability and integrity	
Threat	An attacker submits malicious scripts to the server	The attacker manipulates the input and passes arbitrary data to a target component that Will be processed as normal data. The injected data may include binary code, SQL statements, XML, OS commands etc.
Threat agent	An attacker capable of writing malicious scripts (e.g. SQL and xPath injections)	
Attack method	Submit malicious scripts to the server by acting as a client. Misuse of data: take, read and store data, alter or delete data stored in server. Misuse of business service: make business service unavailable	
Vulnerability	Poor or lack of input filtering at the input interface and server	
Risk treatment	Risk reduction	
Security requirement	Filter input at input interface and server	Filtering, Storage security
Control	Filter input for special characters and keywords, use whitelist of acceptable inputs	Input filtering

Table 24. Conformation of SRP3b and UFP 8.6, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP3b. Secure business activity from buffer overflow attacks	UFP 8.6 Process overflow attack (Uzunov & Fernandez, 2013, Table 5)
Business asset	Data being entered, business service	Messages
IS asset	Input interface, transmission channel, server	User interaction, i.e., input ports Data/storage management, i.e., data structures Distribution control, i.e., software component interfaces, execution abstractions, processes
Security criteria	Integrity and confidentiality of data, integrity and availability of business service	
Impact	Unexpected behaviour, loss of data integrity and confidentiality, loss of business service integrity and availability	
Threat	An attacker triggers a buffer overflow in the system, causing unexpected behaviour or redirection of execution	An attacker injects arbitrary binary code into the process execution environment that will overflow certain pre-defined boundaries and allow control to be passed to that code.
Threat agent	An attacker aware of overflow weakness in software	
Attack method	An attacker triggers an integer overflow by entering an non-conforming input into a business activity	
Vulnerability	Missing bounds checking in software, poor or lack of input filtering at the input interface and server	
Risk treatment	Risk reduction	
Security requirement	Filter input at input interface and server[2] Implement boundary checking[2]	Execution control
Control	Filter input for special character and keywords[2] Use whitelist of acceptable inputs[2] Ensure numeric input is in expected range[2]	Buffer overflow prevention

[2] https://cwe.mitre.org/data/definitions/190.html

Table 25. Conformation of SRP4a and UFP 2.9, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP4a. Secure business services against message flooding attacks	UFP 2.9 Message flooding (Uzunov & Fernandez, 2013, Table 3)
Business asset	Request from client	Business service
IS asset	Transmission channel, server	Communication, i.e., message channels, networking infrastructure
Security criteria	Availability of business service	Availability
Impact	Simultaneous requests keep the server busy and lead to unavailability of business service	
Threat	An attacker performs many simultaneous requests to the server making the server available to normal users	A large stream of messages is injected into a new or existing communication stream to degrade the network performance or interrupt the normal service of a target node or component.
Threat agent	An attacker with an access to a large number of computers and capable of performing simultaneous requests	
Attack method	Create many simultaneous requests to the server thus making the business service unavailable (DDoS attack)	
Vulnerability	Poor or lacking checks for abnormal requests at the server	
Risk treatment	Risk reduction	
Security requirement	Filter abnormal requests at the server	Filtering, Secure communication, Logging and monitoring
Control	Use firewall, DoS Defence System	Network level firewall, IDS

Table 26. Conformation of SRP4b and UFP 8.8, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP4b. Secure business services against resource exhaustion attacks	UFP 8.8 Resource exhaustion (Uzunov & Fernandez, 2013, Table 9)
Business asset	Business service, request from client	
IS asset	Transmission channel, server, input interface	Data/storage management, i.e., storage abstractions, database systems, file systems Resource management Distribution control, i.e., execution abstractions, processes
Security criteria	Availability of business service	Availability
Impact	Simultaneous requests keep the server busy and lead to unavailability of business service	
Threat	An attacker triggers allocation of resource(s) (e.g., memory, file system storage, database connection pool entries, CPU) that leads to unavailability of business service for normal users	A system is saturated with resource requests, or resources are used excessively (e.g. memory hogging applications).
Threat agent	An attacker who is aware of poor or lack of resource restrictions, or a memory leak	
Attack method	Trigger allocation of resources thus slowing down business services response times and/or causing unavailability of business service	
Vulnerability	Software does not properly restrict the size or amount of resources utilized by an actor because of error conditions, exceptional circumstances or confusion over which component is responsible for releasing the resource	
Risk treatment	Risk reduction	
Security requirement	Design throttling mechanisms into the system architecture that limit the amount of resources an unauthorized user can allocate[3] Strong authentication and access control[3] Resource exhaustion detection mechanism that exposes the attack and blocks the attacker[3]	Execution control
Control	Uniformly throttled requests[3]	Quotas and limits

[3] https://cwe.mitre.org/data/definitions/400.html

Table 27. Conformation of SRP5a and UFP 5.1, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat patterns for distributed systems
Pattern	SRP5a. Secure data in data store from corruption	UFP 5.1 Corruption (Uzunov & Fernandez, 2013, Table 6)
Business asset	Data stored in server, business service	Data, business service
IS asset	Data input/output interface, transmission channel, server	Data/storage management, e.g., storage
Security criteria	Data integrity, availability of business service	Integrity of system state, availability of business service
Impact	Loss of data integrity, loss of business service availability, compromise of business	
Threat	An attacker modifies or deletes confidential data	The attacker modifies stored data on persistent or transient storage. The corrupted data can be a cause for corruption of state in the system, or can be used for denial-of-service (e.g. if the data is sensitive configuration data that is rendered useless).
Threat agent	A malicious insider misusing access to the server and manipulating confidential data	
Attack method	Modify or delete confidential data by using access to the disk, causing loss of data integrity and possibly loss of business service availability	
Vulnerability	Poor or lack of data integrity checks	
Risk treatment	Risk reduction	
Security requirement	Implement database integrity defence mechanism with dual level signatures (Barbará *et al.*, 2000)	Storage security
Control	Monitor data access, monitor and verify database signature changes (Barbará *et al.*, 2000)	Hashing, integrity checks, secure backup

Table 28. Conformation of SRP5b and UFP 7.1, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat patterns for distributed systems
Pattern	SRP5b. Secure the integrity of auditing information	UFP 7.1 Track erasing (Uzunov & Fernandez, 2013, Table 8)
Business asset	Data stored in server	Auditing information
IS asset	Data input/output interface, transmission channel, server	Data/storage management, i.e., database systems, file systems
Security criteria	Data integrity	Integrity
Impact	Loss of data integrity, compromise of business	
Threat	An attacker modifies or deletes confidential auditing data	An attacker modifies or deletes auditing information to erase the tracks of an attack.
Threat agent	A malicious insider misusing access to the server and manipulating confidential data	
Attack method	Perform malicious operations on the data in database. After attack edit or delete any auditing information that might trace the attack	
Vulnerability	Poor or lack of data integrity checks	
Risk treatment	Risk reduction	
Security requirement	Implement database integrity defence mechanism with dual level signatures (Barbará *et al.*, 2000)	Logging and monitoring
Control	Monitor data access, monitor and verify database signature changes (Barbará *et al.*, 2000)	Secure system logs, replicated logs

Table 29. Conformation of SRP5c and UFP 8.1, adapted from (Ahmed & Matulevičius, 2013) and (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat patterns for distributed systems
Pattern	SRP5c. Secure data in data store from unauthorized access	UFP 8.1 Unauthorized access (Uzunov & Fernandez, 2013, Table 9)
Business asset	Data stored in server	Data
IS asset	Data input/output interface, transmission channel, server	User interaction, e.g., input ports Resource management, e.g., resources Data/storage management, e.g., storage abstractions, database systems, file systems
Security criteria	Data confidentiality	
Impact	Loss of data confidentiality, compromise of business	
Threat	An attacker gets hold of confidential data	A user gains access to data or resources without proper authorization
Threat agent	A malicious insider misusing access to the server and retrieving confidential data	
Attack method	Retrieve confidential data from the server using assigned credentials. Exploit data: read and save for personal use	
Vulnerability	Poor or lack of data invisibility	
Risk treatment	Risk reduction	
Security requirement	Ensure invisibility of confidential data	Authorization
Control	Encryption algorithm, monitor data access	

Table 30. Conformation of SRP6 and UFP 6.3, adapted from (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP6. Prevent harm to business service through leakage of confidential information via error and standard response messages	UFP 6.3 Output information disclosure (Uzunov & Fernandez, 2013, Table 7)
Business asset	Business service	Information
IS asset	Input interface, Server	User interaction (i.e. output ports) Distribution control (i.e. all aspects related to component collaboration or interaction, coordination of local/remote execution and synchronization)
Security criteria	Availability of business service, integrity and confidentiality of data	Confidentiality
Impact	Compromise of business, unavailability of business service, integrity and confidentiality of data	
Threat	An attacker intentionally causes the system to produce an error or a standard response message which contains sensitive information from the system (e.g. system configuration, credentials or private information) that the attacker can use to launch new attacks	Information from the system is leaked to the attacker via some form of output, such as error messages or standard responses
Threat agent	An attacker who is able to trigger a leakage of confidential information from the system through an error or standard response message and use this information to launch new attacks	
Attack method	Trigger an error or a standard response message from the system. Use gained information to launch new attacks.	
Vulnerability	Server does not filter confidential information from error messages/standard responses	
Risk treatment	Risk reduction	
Security requirement	Filter confidential information from error messages/standard responses[4] Configure server software so that confidential information does not leak[4]	Filtering
Control	Parse server messages before passing to user[4] Disable debug messages[4] Specify error messages such that no sensitive data is sent[4] Capture all exceptions and return generic error messages[4] Use default error messages or error pages that do not leak information[4]	Output limitation

[4] https://cwe.mitre.org/data/definitions/209.html

Table 31. Conformation of SRP7 and UFP 8.9, adapted from (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP7. Protect business service from deadlock attacks	UFP 8.9 Targeted process crashing (Uzunov & Fernandez, 2013, Table 9)
Description	This pattern secures the system from deadlock attacks where multiple processes are initiated that become dependent on one-another to finish, causing the service to become unavailable for regular users.	
Business asset	Request from client, business service	Process
IS asset	Input interface, Server	Resource management, e.g., resources Distribution control, e.g., execution abstractions, processes
Security criteria	Availability of business service	Availability
Impact	Unavailability of business service	
Threat	An attacker triggers a deadlock situation and causes unavailability of the business service	A process is purposefully exploited to crash, losing availability for the rest of the system
Threat agent	An attacker with knowledge of deadlock situations at the server and an intention to trigger them	
Attack method	Trigger two or more competing processes that will wait for each other to finish	
Vulnerability	Simultaneous processes are not synchronized at the server	
Risk treatment	Risk reduction	
Security requirement	Synchronize the execution and operation of simultaneous processes[5]	Execution control
Control	Use non-blocking synchronization algorithms and/or libraries that implement synchronization[5]	Process replication

[5] https://capec.mitre.org/data/definitions/25.html

Table 32. Conformation of SRP8a and UFP 9.4, adapted from (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP8a. Secure the system from brute force password attacks	UFP 9.4 Password attacks (guessing, brute force, rainbow tables, etc.) (Uzunov & Fernandez, 2013, Table 10)
Business asset	Data, business services	Passwords
IS asset	Input interface, server	
Security criteria	Confidentiality and integrity of data Availability of business services	
Impact	Loss of data confidentiality and integrity, loss of business service's availability	
Threat	An attacker with knowledge of at least one username in the system uses a dictionary-based or an alphabet-based brute force attack to gain access to the system	An attacker acquires passwords easily (e.g. passed as plaintext) or takes advantage of incorrect password storage (lack of salting and stretching).
Threat agent	An attacker with knowledge of at least one username in the system	
Attack method	An attacker tries each entry in a dictionary or any possible value using a given alphabet as password to gain access to the system using an existing user account	
Vulnerability	System uses one factor password based authentication System has weak or no password policy System does not implement password throttling mechanism	
Risk treatment	Risk reduction	
Security requirement	Block user after certain number of failed login attempts[6]	
Control	Password throttling mechanism[6] Enforce strong password policy to ensure users use strong passwords[6] Enforce users to change passwords after a given time period[6]	

[6] https://capec.mitre.org/data/definitions/49.html

Table 33. Conformation of SRP8b and UFP 10.1, adapted from (Uzunov & Fernandez, 2013)

	Security risk-oriented pattern	Threat pattern for distributed systems
Pattern	SRP8b. Protect system from unauthorized access through common or default usernames and passwords	UFP 10.1 Use of default credentials (Uzunov & Fernandez, 2013, Table 11)
Business asset	Data, user accounts	
IS asset	Input interface, Server	
Security criteria	Integrity and confidentiality of data, availability of business service	
Impact	Loss of data confidentiality, compromise of business	
Threat	An attacker accesses the system using a common/default username and password and performs unauthorized actions	An attacker uses default credentials for access or authentication.
Threat agent	An attacker with a collection of vendor default credentials and/or common usernames and passwords	
Attack method	Try common and/or default usernames and passwords to gain access to the system	
Vulnerability	System uses one factor password based authentication and there exist vendor default account credentials	
Risk treatment	Risk reduction	
Security requirement	Block user after certain number of failed login attempts[7] Deny access if user tries to authenticate using a default account[7]	Identity management
Control	Delete all vendor account credentials from the system[7] Implement strong password policy[7] Make users change passwords after a set time period[7] Implement invalid login throttling mechanism[7]	

[7] https://capec.mitre.org/data/definitions/70.html

Table 34. SRP8c: Securing the system from account lockout attacks

	Security risk-oriented pattern
Pattern	SRP8c. Secure the system from account lockout attacks
Description	This is a sub-pattern of SRP8a which prevents the exploitation of security mechanisms used for securing the system from brute force attacks.
Business asset	Business service
IS asset	Input interface, server
Security criteria	Availability of business service for specific user(s)
Impact	Unavailability of business service for legitimate user(s)
Threat	An attacker locks out legitimate users' accounts, e.g. by entering invalid login credentials at the input interface for a given amount of times, thus exploiting the system's lockout mechanism
Threat agent	An attacker with knowledge of at least one legitimate username
Attack method	The attacker behaves in a manner that results with a legitimate user being locked out of their own account (e.g. enters the wrong password for a given amount of times)
Vulnerability	The system has an account lockout mechanism
Risk treatment	Risk reduction
Security require-ment	Detect account lockout attack and block the attacker[8]
Control	Intelligent password throttling mechanism that is able to detect account lockout attacks and block the attacker (e.g. based on IP address)[8]

[8] https://capec.mitre.org/data/definitions/2.html

III. Application of Security Risk-oriented Patterns to Five Example Business Processes

Passenger Check-in Process

Below are rest of the risk-oriented and security requirements models resulting from the application of the SRPs to the passenger check-in process. The models are grouped according to the SRPs that are applied in them.

SRP2b

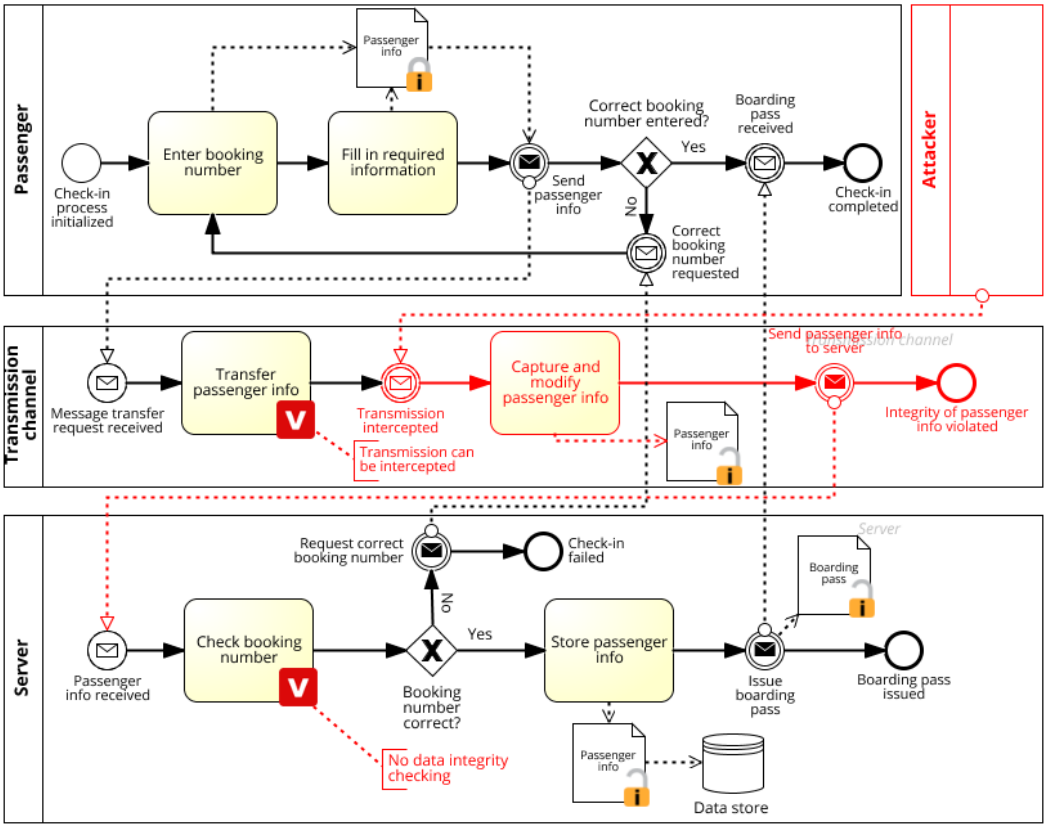


Figure 60. SRP2b: Security Risk-oriented Process Model (Business Asset: Passenger Info)

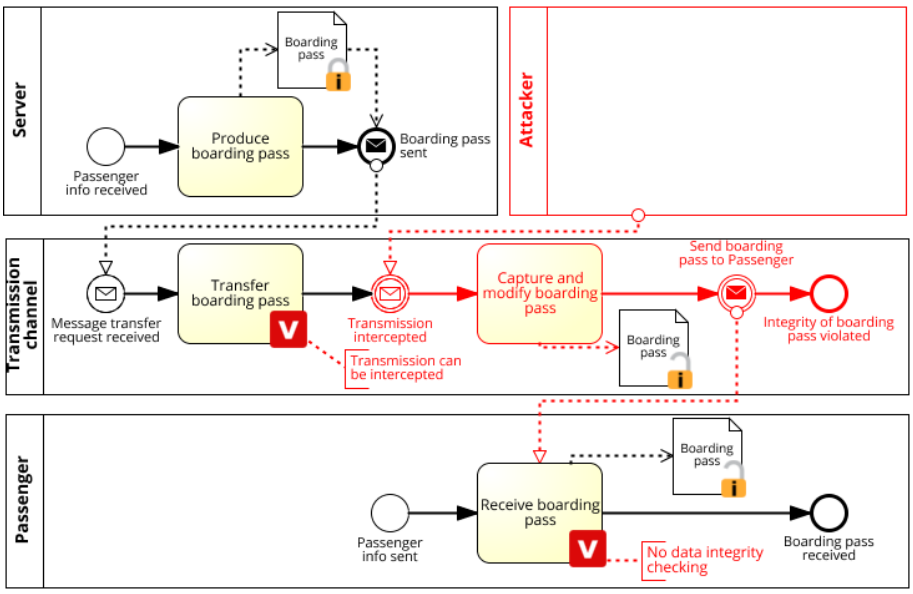


Figure 61. SRP2b: Security Risk-oriented Process Model (Business Asset: Boarding Pass)

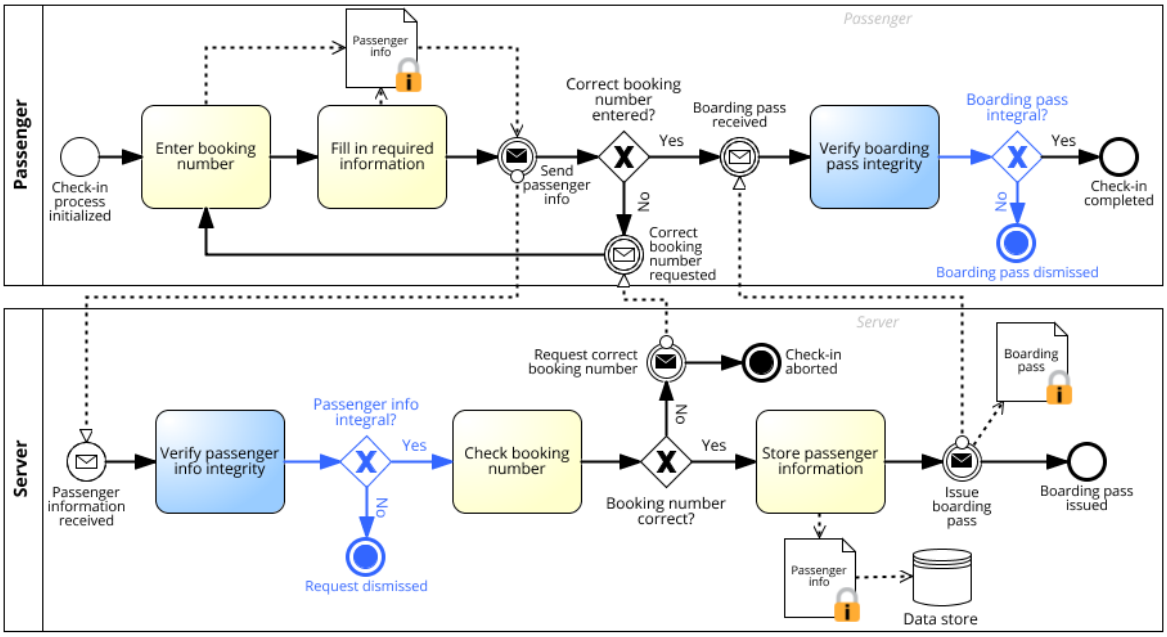


Figure 62. SRP2b: Security Requirements for Business Process

SRP3a & SRP3b

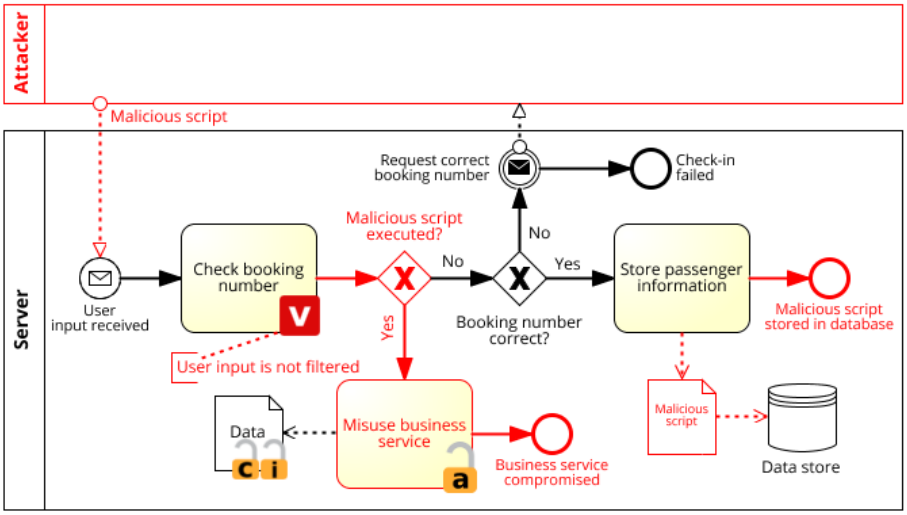


Figure 63. SRP3a & SRP3b: Security Risk-oriented Process Model

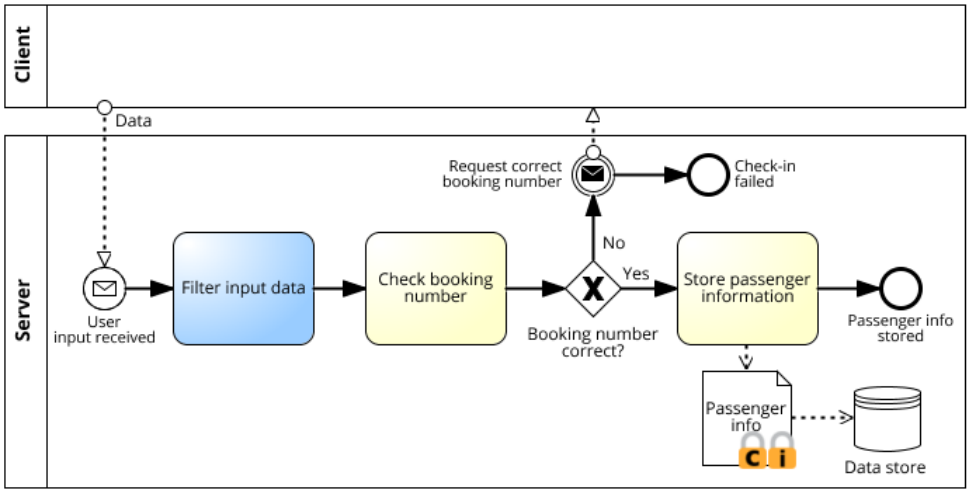


Figure 64. SRP3a & SRP3b: Security Requirements for Business Process

SRP4a

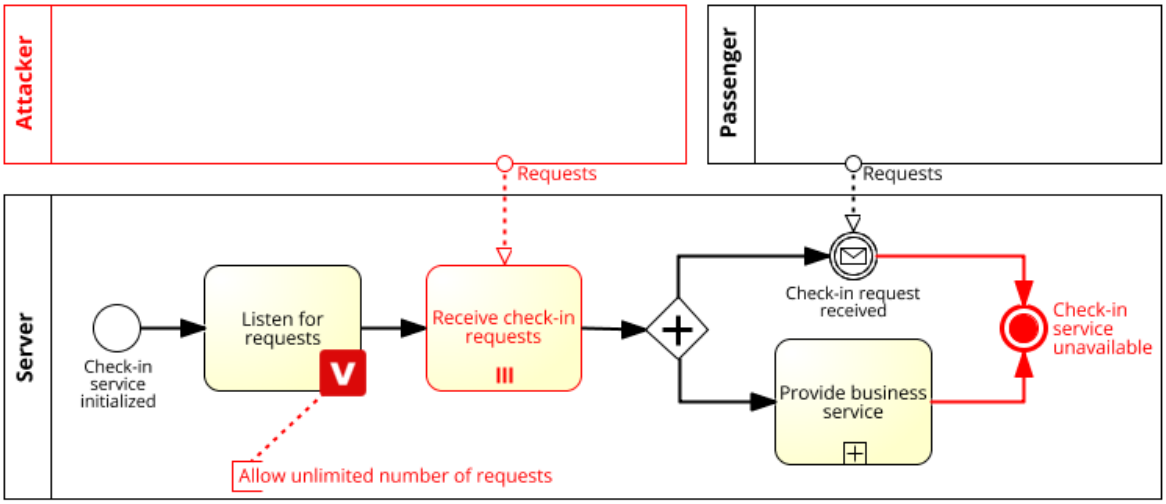


Figure 65. SRP4a: Security Risk-oriented Process Model

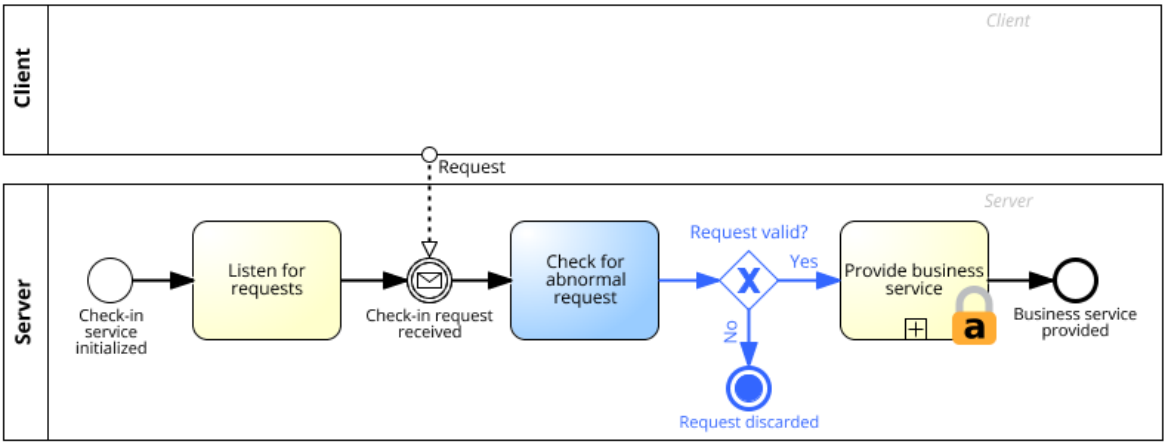


Figure 66. SRP4a: Security Requirements for Business Process

SRP5a

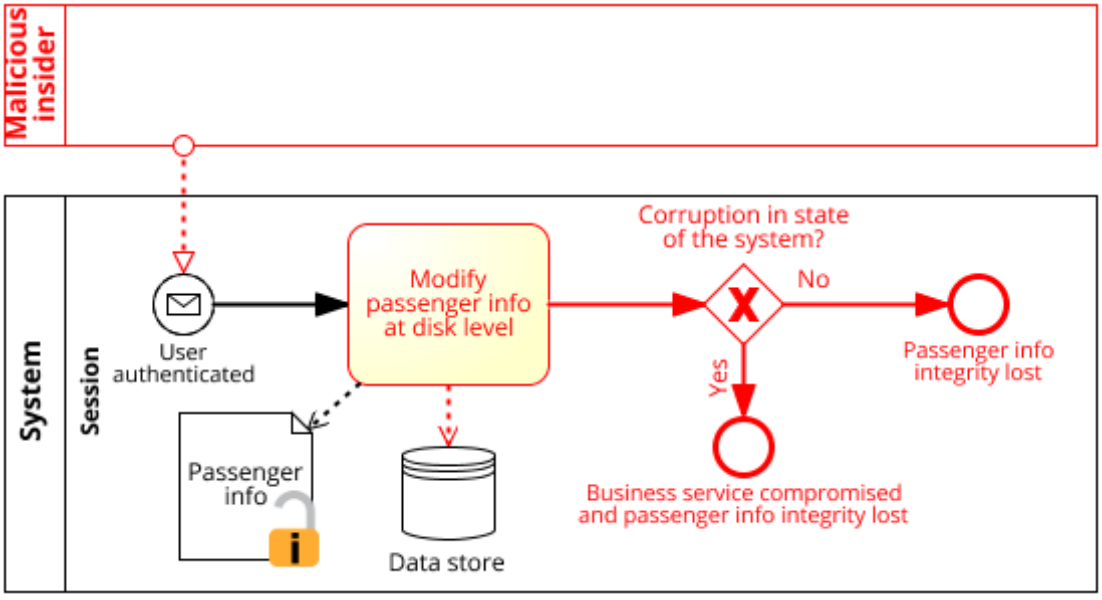


Figure 67. SRP5a: Security Risk-oriented Process Model

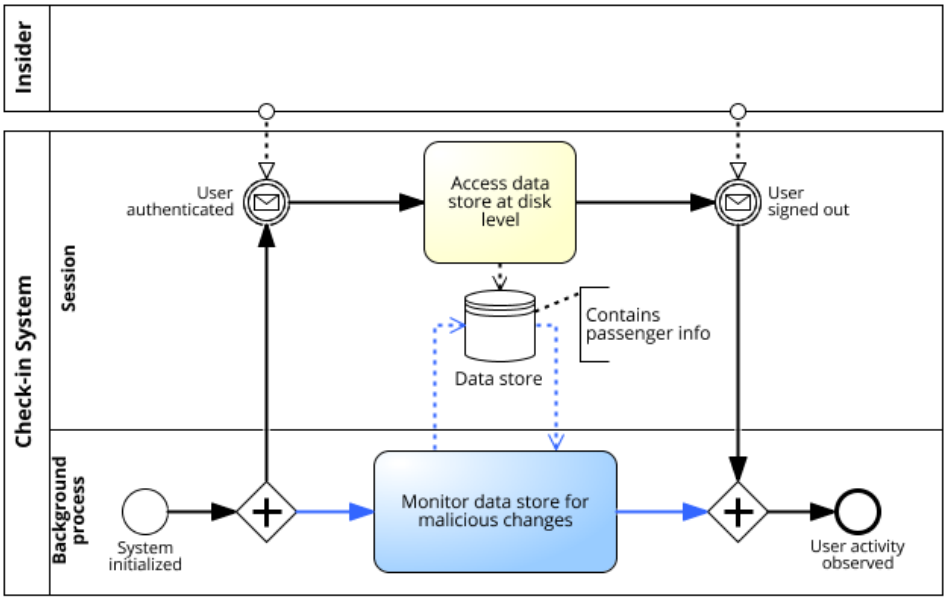


Figure 68. SRP5a: Security Requirements for Business Process

SRP5b

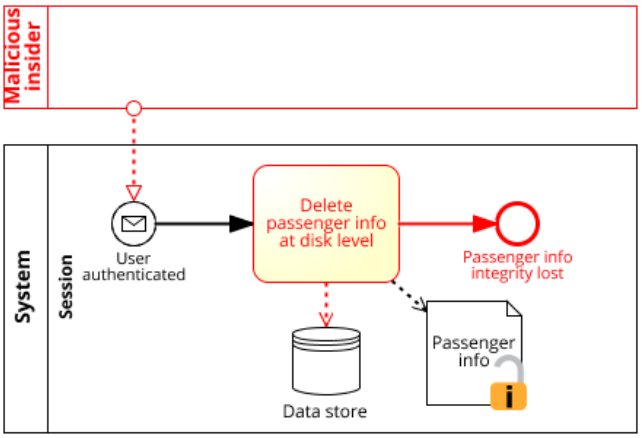


Figure 69. SRP5b: Security Risk-oriented Process Model

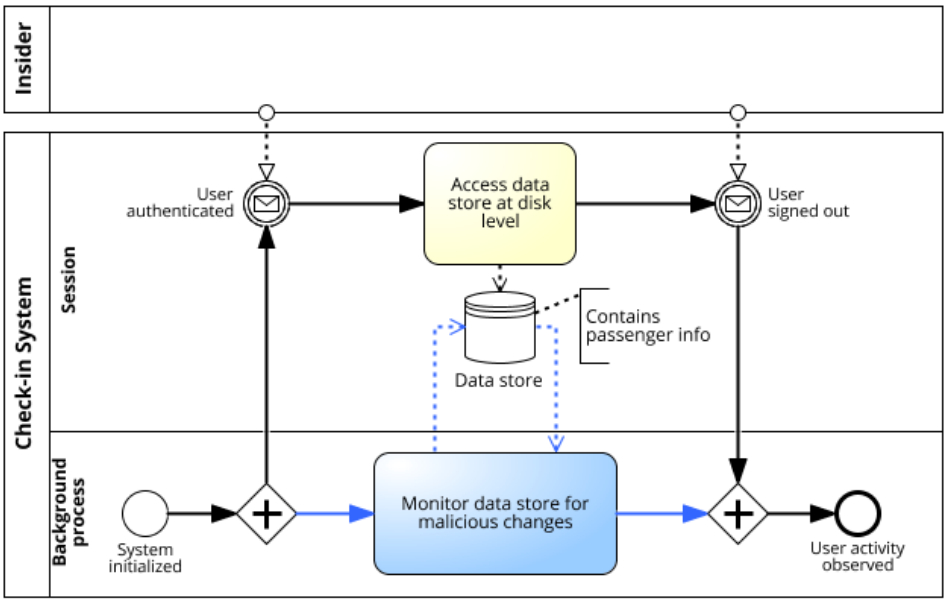


Figure 70. SRP5b: Security Requirements for Business Process

SRP5c

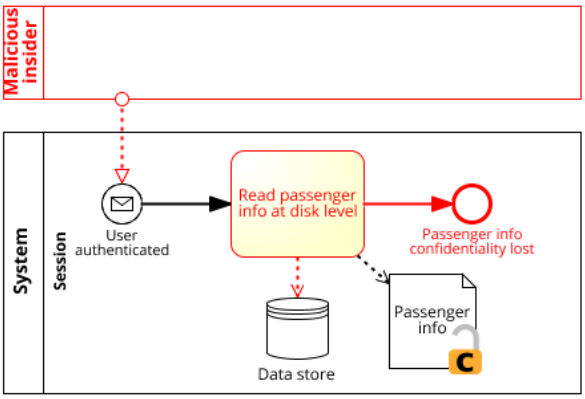


Figure 71. SRP5c: Security Risk-oriented Process Model

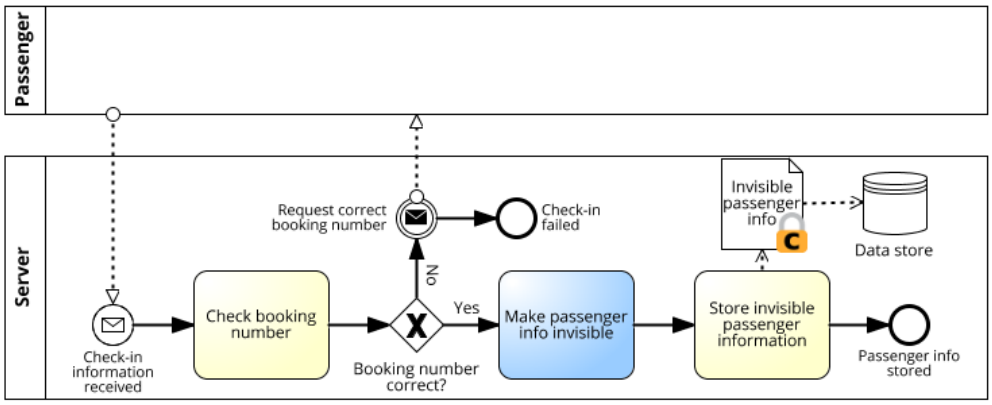


Figure 72. SRP5c: Security Requirements for Business Process

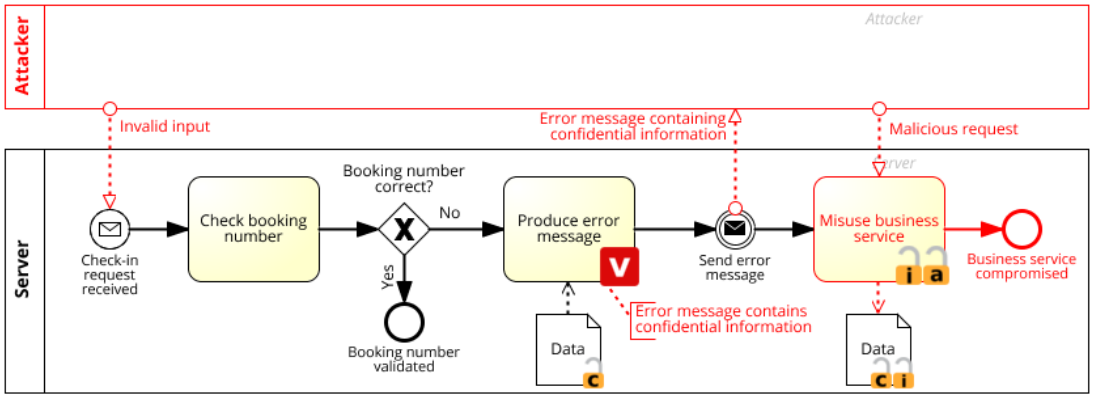


Figure 73. SRP6: Security Risk-oriented Process Model

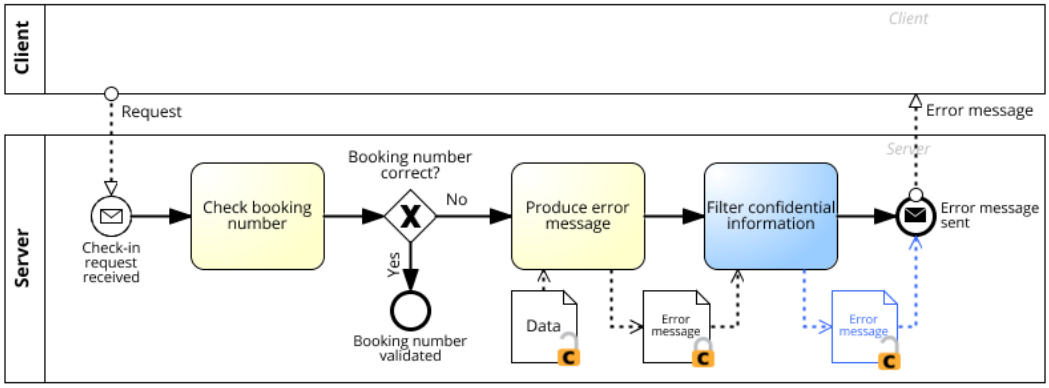


Figure 74. SRP6: Security Requirements for Business Process

Securing Baggage Check-in Process

Figure 75 depicts the baggage check-in process. This example process is based on Brussels Airlines baggage check-in service[9]. The passenger enters their booking number, fills in the required information (e.g., number of bags) and submits it to the server. The server verifies the submitted information, stores it in a data store and issues bag tags for the passenger.

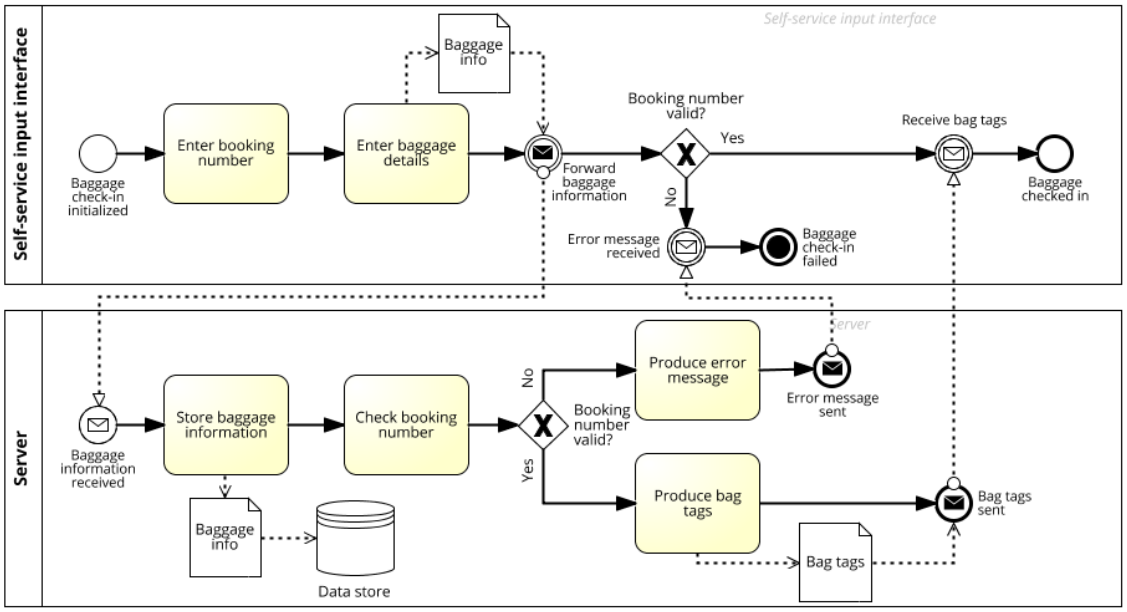


Figure 75. Baggage Check-in Process

Figure 76 presents a BPMN model of the final security requirements for the baggage check-in process. Table 35 contains the security requirements for the baggage check-in process together with respective control methods.

Table 35. Security Requirements for Baggage Check-in Process

Req. ID	Security Requirement	Control
M2.SRP2a.1	Make baggage info unreadable to attackers during transmission between client and server	Encryption algorithm
M2.SRP2a.2	Make bag tags unreadable to attackers during transmission between client and server	
M2.SRP2b.1	Verify integrity of baggage info after transmission	Checksum algorithm
M2.SRP2b.2	Verify integrity of bag tags after transmission	
M2.SRP4a.1	Filter abnormal requests at the server	Firewall, DoS Defence System
M2.SRP3a.1	Filter input after receiving baggage check-in request	Filter input for special characters and keywords, use whitelist of acceptable inputs
M2.SRP5a.1	Monitor the data store at the server for malicious changes	Monitor and verify database signature changes
M2.SRP6.1	Filter confidential information from error messages and standard responses	Disable debug messages, use default error messages or error pages that do not leak information
M2.SRP5c.1	Make baggage info invisible before storing in data store	Encryption algorithm, monitor data access

[9] https://www.brusselsairlines.com/en-se/misc/home-printed-baggage-tags.aspx

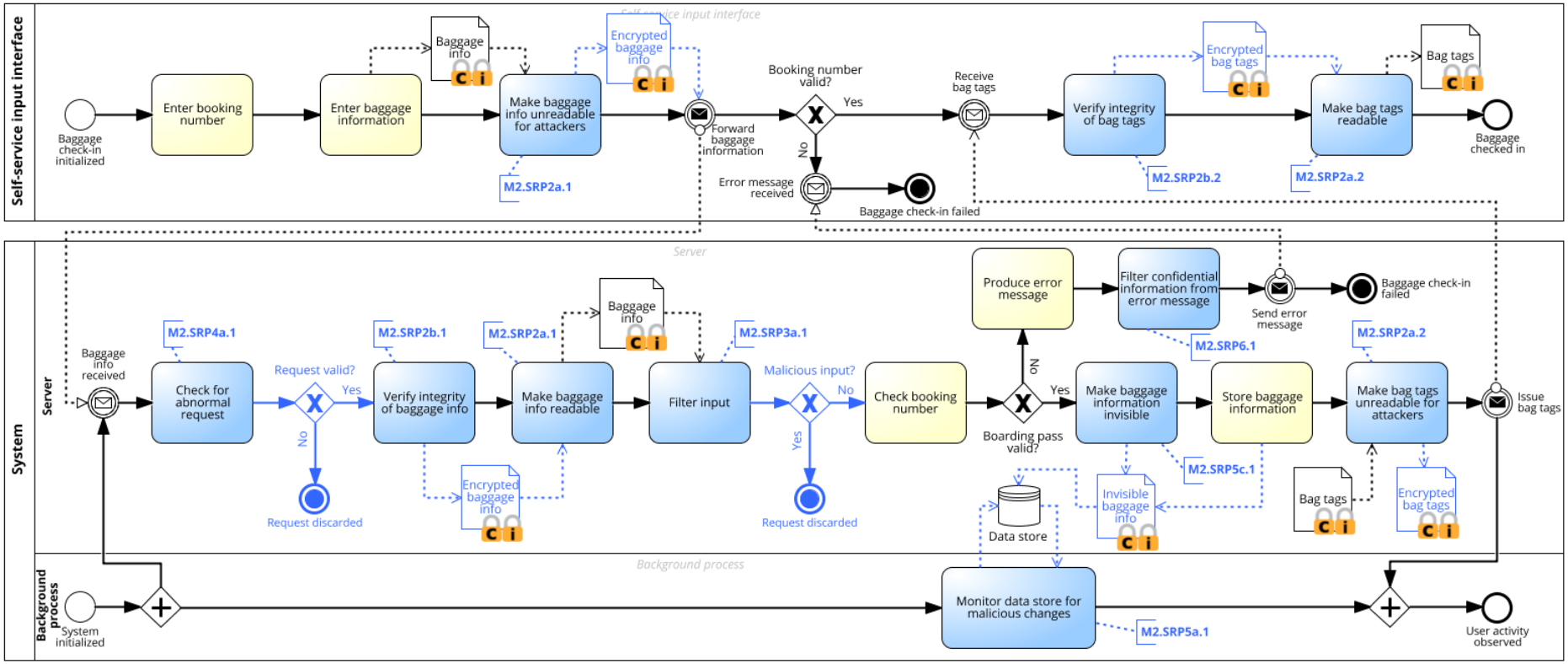


Figure 76. Security Requirements for Baggage Check-in Process

Securing Fuel Service Form Issuing Process

Figure 77 depicts the fuel service form issuing process. This example is based on flight planning process descriptions shared by Airline Dispatchers Federation[10] and a major U.S. airline pilot[11]. The flight dispatcher calculates the fuel quantity based on numerous variables such as flight distance, aircraft type, weather conditions etc. This information is sent to the server where it is stored and in turn sent to the pilot. The pilot checks the quantity and increases it if necessary. A fuel service form is issued then and stored in the server's data store.

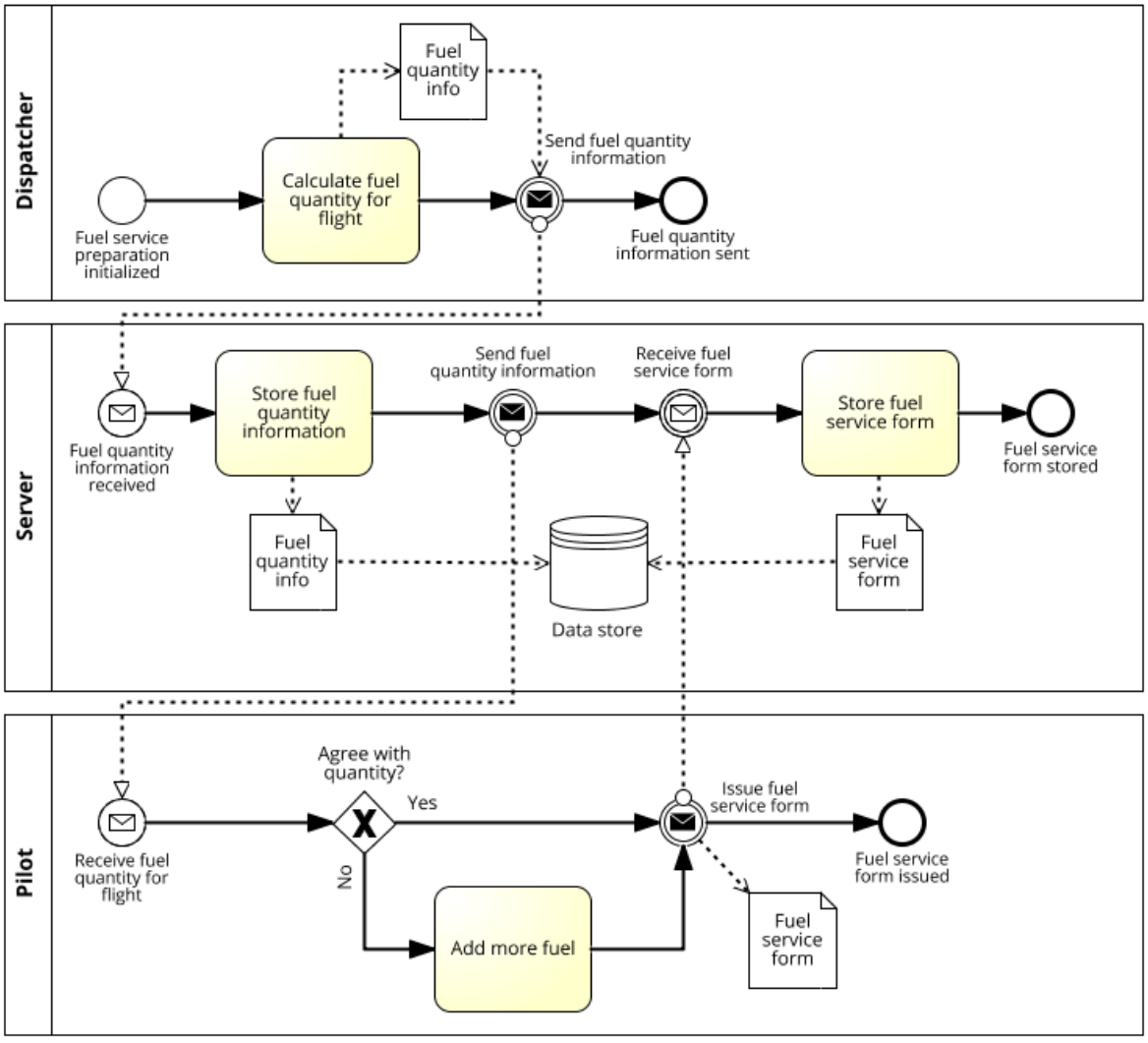


Figure 77. Fuel Service Form Issuing Process

Figure 78 presents a BPMN model of the final security requirements for the fuel service form issuing process. Table 36 contains the security requirements for the fuel service form issuing process together with respective control methods.

[10] https://www.dispatcher.org/dispatcher/job-description
[11] http://www.myairlineflight.com/fueling.html

Table 36. Security Requirements for Fuel Service Form Issuing Process

Req. ID	Security Requirement	Control
M3.SRP2a.1	Make fuel quantity info unreadable to attackers during transmission between dispatcher and server	Encryption algorithm
M3.SRP2a.2	Make fuel quantity info unreadable to attackers during transmission between pilot and server	
M3.SRP2a.3	Make fuel service form unreadable to attackers during transmission between pilot and server	
M3.SRP2b.1	Verify integrity of fuel quantity information after transmission from dispatcher to server	Checksum algorithm
M3.SRP2b.2	Verify integrity of fuel quantity information after transmission from server to pilot	
M3.SRP2b.3	Verify integrity of fuel service form after transmission from pilot to server	
M3.SRP4a.1	Filter abnormal requests at the server	Firewall, DoS Defence System
M3.SRP3a.1	Filter input after receiving fuel quantity information at the server	Filter input for special characters and keywords, use whitelist of acceptable inputs
M3.SRP5c.1	Make fuel quantity info invisible before storing in data store	Encryption algorithm, monitor data access
M3.SRP5c.1	Make fuel service form invisible before storing in data store	
M3.SRP5a.1	Monitor the data store at Server for malicious changes	Monitor and verify database signature changes

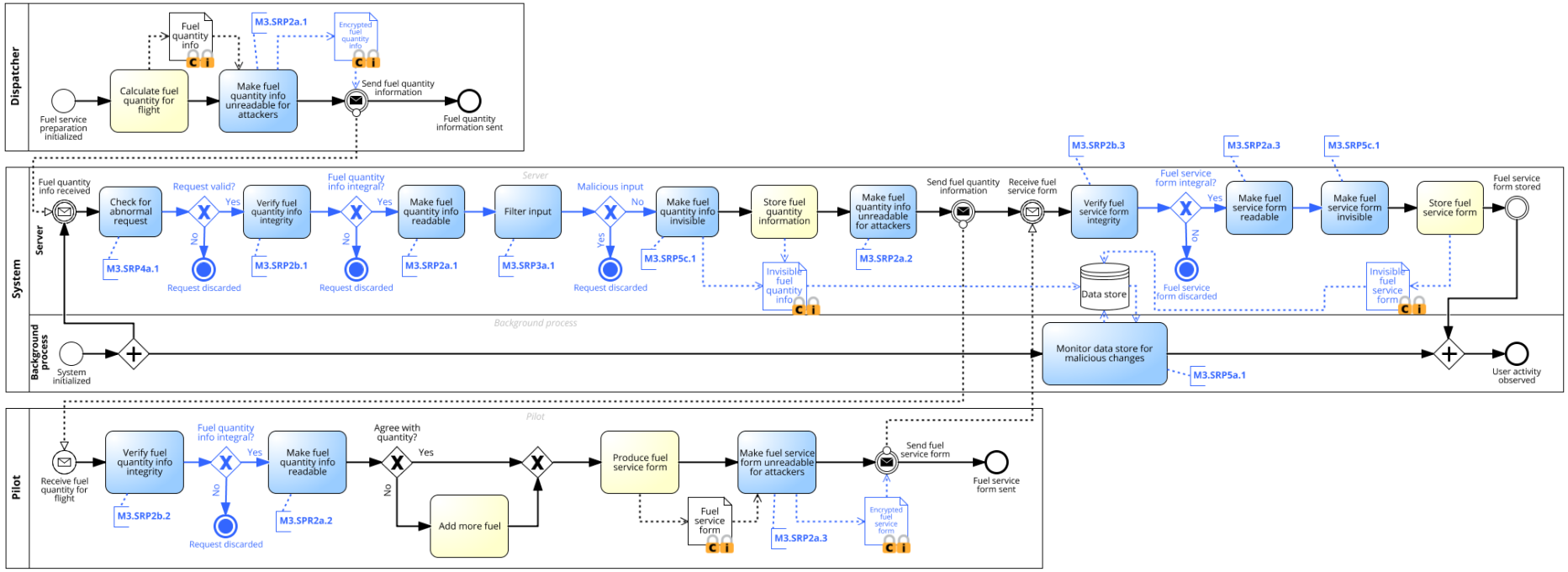


Figure 78. Security Requirements for Fuel Service Form Issuing Process

Securing Fuel Service Form Requesting Process

Figure 79 depicts the fuel service form requesting process. This example is based on flight planning process descriptions by a major U.S. airline pilot[12]. In this process a fuel service form is requested using an input/output interface. The server receives the request, queries the data store for the requested information and returns it to the input/output interface. Fuelling service can then be performed according to the quantity and fuel type information in the form.

Figure 80 presents a BPMN model of the final security requirements for the fuel service form requesting process. Table 37 contains the security requirements for the fuel service form requesting process together with respective control methods.

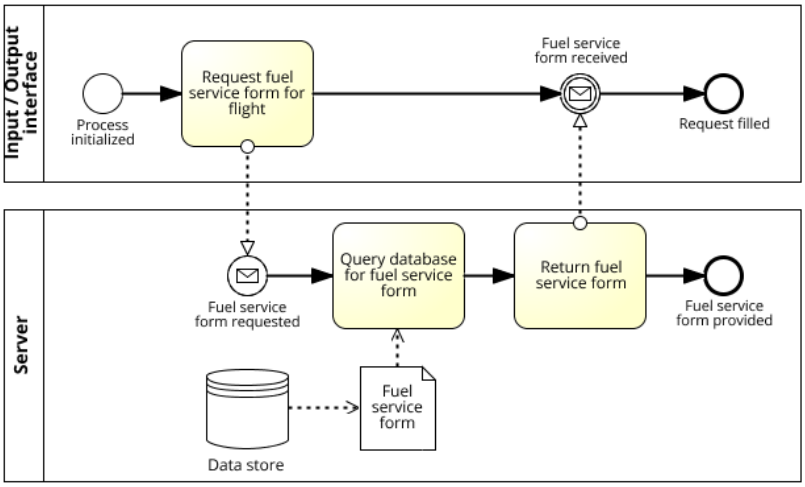


Figure 79. Fuel Service Form Requesting Process

Table 37. Security Requirements for Fuel Service Form Requesting Process

Req. ID	Security Requirement	Control
M4.SRP2a.1	Make fuel service form request unreadable to attackers during transmission between input/output interface and server	Encryption algorithm
M4.SRP2a.2	Make fuel service form unreadable to attackers during transmission between input/output interface and server	
M4.SRP2b.1	Verify integrity of fuel service form request after transmission from input/output interface to server	Checksum algorithm
M4.SRP2b.2	Verify integrity of fuel service form after transmission from server to input/output interface	
M4.SRP4a.1	Filter abnormal requests at the server	Firewall, DoS Defence System
M4.SRP3a.1	Filter input after receiving fuel service form request at the server	Filter input for special characters and keywords, use whitelist of acceptable inputs
M4.SRP5c.1	Make fuel service form visible after querying from data store	Cryptographic algorithm
M4.SRP5a.1	Monitor the data store at Server for malicious changes	Monitor and verify database signature changes
M4.SRP1a.1	Check access rights before querying data store for fuel service form	Define user levels with appropriate access authorization with credentials, monitor data access

[12] http://www.myairlineflight.com/fueling.html

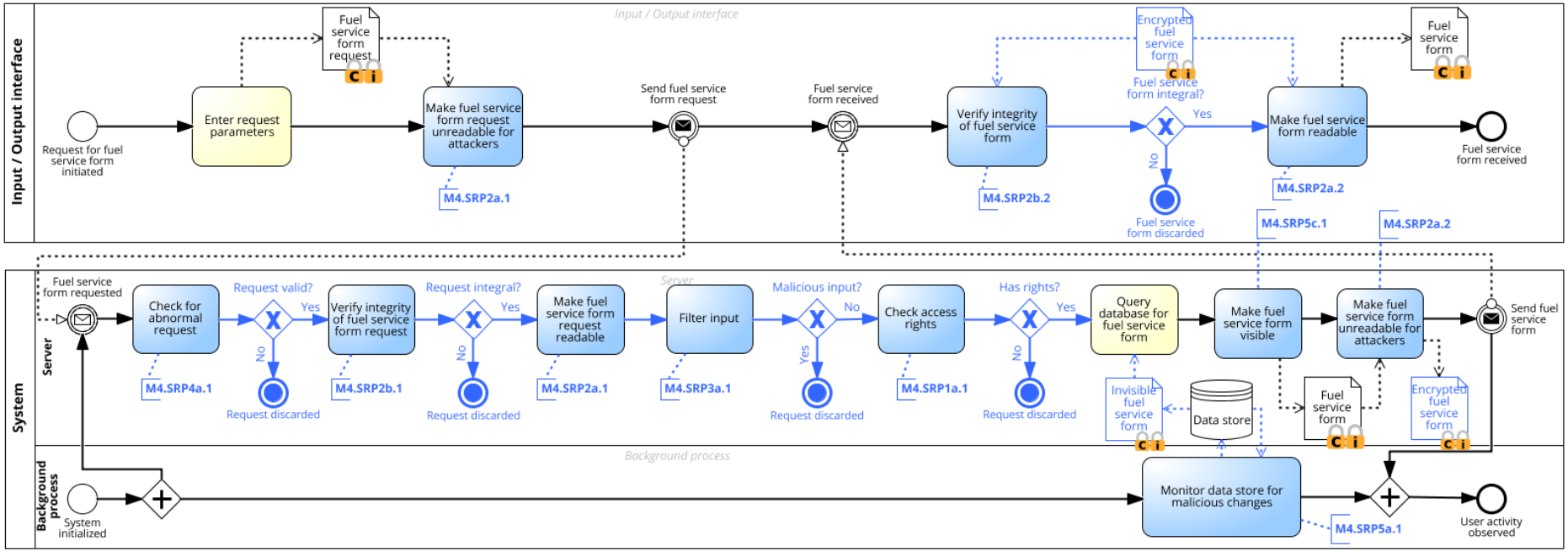


Figure 80. Security Requirements for Fuel Service Form Requesting Process

Securing Loading Instruction Form Requesting Process

Figure 81 depicts the loading instruction form requesting process which is one of the sub-processes of airline ground operations. The example is based on SKYbrary's[13] (a repository of knowledge related to aviation processes) descriptions of the aircraft loading process. A loading instruction form that describes the load disposition in the aircraft is produced either by a dispatcher or an automated system. In order to begin loading the aircraft, a loading supervisor needs to request this information from the server. The server queries the requested loading instruction form from a data store and returns it to the loading supervisor.

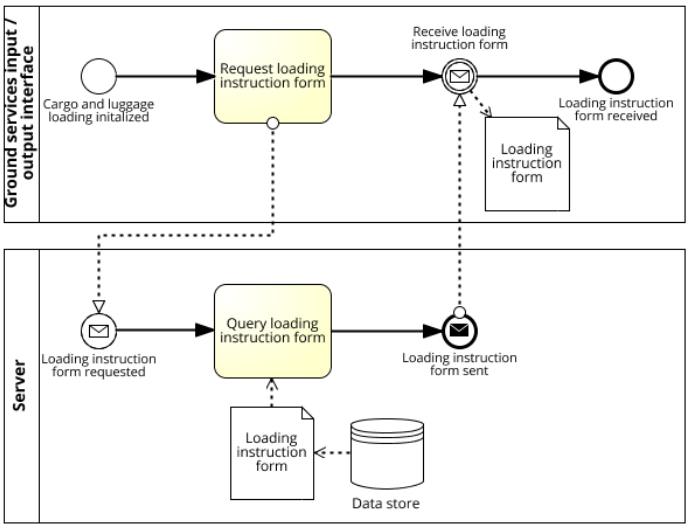


Figure 81. Loading Instruction Form Requesting Process

Figure 82 presents a BPMN model of the final security requirements for the loading instruction form requesting process. Table 38 contains the security requirements for the loading instruction form requesting process together with respective control methods.

Table 38. Security Requirements for Loading Instruction Form Requesting Process

Req. ID	Security Requirement	Control
M5.SRP2a.1	Make loading instruction form request unreadable to attackers during transmission between input/output interface and server	Encryption algorithm
M5.SRP2a.2	Make loading instruction form unreadable to attackers during transmission between input/output interface and server	
M5.SRP2b.1	Verify integrity of loading instruction form request after transmission from input/output interface to server	Checksum algorithm
M5.SRP2b.2	Verify integrity of loading instruction form after transmission from server to input/output interface	
M5.SRP4a.1	Filter abnormal requests at the server	Firewall, DoS Defence System
M5.SRP3a.1	Filter input after receiving loading instruction form request at the server	Filter input for special characters and keywords, use whitelist of acceptable inputs
M5.SRP5c.1	Make loading instruction form visible after querying from data store	Cryptographic algorithm
M5.SRP5a.1	Monitor the data store at Server for malicious changes	Monitor and verify database signature changes
M5.SRP1a.1	Check access rights before querying data store for loading instruction form	Define user levels with appropriate access authorization with credentials, monitor data access

[13] http://www.skybrary.aero/index.php/Loading_of_Aircraft_Holds

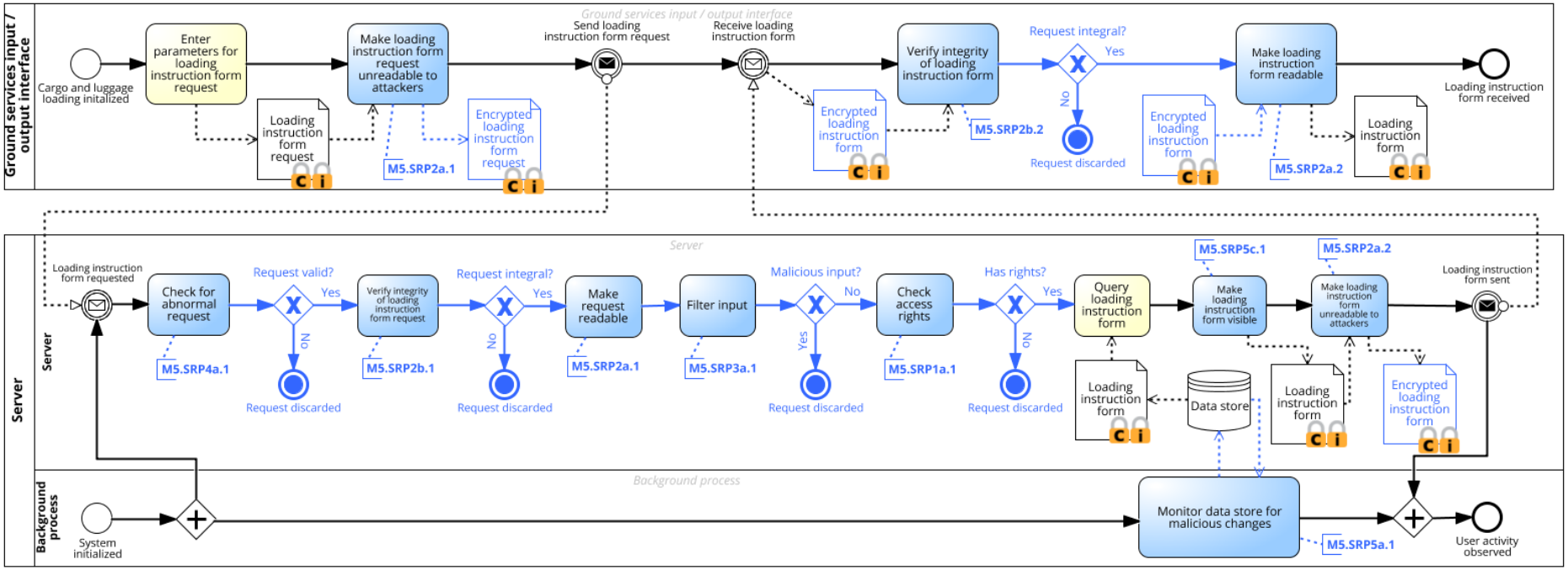


Figure 82. Security Requirements for Loading Instruction Form Requesting

IV. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Silver Samarütel** (date of birth: 22.02.1991),

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

    1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

    1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

Revision of Security Risk-oriented Patterns for Distributed Systems,

supervised by Raimundas Matulevičius,

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **19.05.2016**