University of Exeter
Department of Computer Science

Improving Confidentiality in Inter-Organizational Collaborations

Sakine Yalman

Submitted by Sakine Yalman to the University of Exeter as a thesis for the degree of *Doctor of Philosophy* in *Computer Science*
March 2023

I would like to dedicate this thesis to my loving parents . . .

Abstract

In today's interconnected world, collaborations between organizations have become increasingly important to achieve success. Data sharing plays an important role in these collaborations as it enables participants to exchange information and make informed decisions together. With the convergence of several technologies, such as the internet of things (IoT), artificial intelligence (AI), machine learning (ML), and cloud computing, collaborations are now possible across multiple domains and organizations.

However, the current working model for inter-organizational collaborations often requires participants to either share data openly between them or at least with a third party that they all have to trust. While data sharing is essential for effective collaborations, in many cases, this open sharing raises concerns regarding business-critical data, causing companies to reconsider their participation in such collaborations. Companies may be reluctant to participate in collaborations due to the risk of compromising their intellectual property, competitive advantage, or trade secrets.

To address this challenge, one potential solution is to allow for secure and confidential computation in collaborations without the need for sharing data in plaintext. Although secure multi-party computation (SMPC) offers a theoretical solution, it is often not practical due to its high computational cost and slow execution. This thesis proposes a scalable privacy-preserving model for inter-organizational collaborations that ensures the confidentiality of shared data without imposing a significant performance overhead. The model is based on a combination of a hierarchical grouping approach and the use of SMPC. Specifically, the model aims to decompose the computation required for collaboration using the hierarchical grouping approach and apply SMPC to the decomposed computation. This approach enables organizations to collaborate with enhanced confidentiality while also being fast enough to enable new ways of collaboration. Additionally, to encourage organizations to join collaborations, it is necessary to examine the security and privacy concerns that come with sharing data. Therefore, this thesis also presents a novel threat modelling approach tailored to inter-organizational collaborations. This threat modelling approach enables organizations to identify potential threats associated with participating in such collaborations and offers systematic guidance on developing mitigation strategies, as well as designing and establishing secure and privacy-friendly collaborations. The whole approach represented in this thesis is applied to two scenarios, lead time and life-cycle assessment, as case studies to be assessed and evaluated. Experimental results show that our approach results in a significant performance gain in most cases where the collaboration structure allows for hierarchical groupings and parallel computations, while in others, it performs at least as well as approaches using the direct application of SMPC. Furthermore, it offers additional security properties beyond these approaches.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Achim D. Brucker, for his valuable guidance and support throughout my research. His expertise, constructive criticism, and valuable feedback have been essential in shaping my research and thesis. I would also like to extend my appreciation to Dr. Jia Hu for providing constructive feedback and asking insightful questions during our annual reports. Moreover, I am deeply grateful to my viva panel, consisting of Prof. Chunbo Luo and Dr. Bo Zhou, for their insightful comments and constructive feedback that have contributed to the refinement of this thesis.

I would also like to express my gratitude to my sponsor, the Ministry of National Education of Türkiye, for granting me a scholarship to pursue my PhD studies. Without their support, I would not have had the opportunity to pursue this degree and conduct research that I am truly passionate about.

Completing a PhD is a challenging and demanding journey, and I am thankful for the friends and fellow researchers who stood by me, supported me, and motivated me along the way. I would like to specifically thank my dear friends Hüseyin Burak Akyol, Mücahit Aydemir, and Meltem Çatalbaş for their solidarity and support throughout my life in the United Kingdom. Their friendship and support have been invaluable to me.

Finally, I am deeply grateful to my beloved parents, Yüksel and Yakup, who have always been my unwavering source of support and encouragement. I thank them for always being there for me, for respecting every decision I have made for my life, and for their unconditional love. This thesis is dedicated to them. I am also grateful to my siblings, Hüseyin, Gülhan, and Muhammet, for standing by me and supporting me every step of the way. Thank you so much, love you all ♡

Contents

Abstract 3

Acknowledgements 5

List of Figures 10

List of Tables 12

Acronyms 13

1 Introduction 15

1.1 Motivation . . . 15
1.2 Contributions . . . 18
1.3 Thesis Structure . . . 19

2 Background 21

2.1 Collaborative Business Networks . . . 21
2.1.1 Supply Chain Management . . . 22
2.1.2 Information Sharing in Supply Chains . . . 22
2.2 Public Key Infrastructure (PKI) . . . 24
2.2.1 Components of a PKI . . . 24
2.2.2 Core Security Service of PKI: Authentication . . . 26
2.2.3 A Simple PKI Implementation Example . . . 26
2.3 Secure Multi-Party Computation (SMPC) . . . 28
2.3.1 Security of SMPC . . . 29
2.3.2 Feasibility of SMPC . . . 32
2.3.3 A Simple SMPC Example . . . 32
2.4 Running Example . . . 34
2.4.1 Determination of Lead Times . . . 34
2.4.2 Bicycle Supply Chain and Lead Time Computation . . . 35
2.5 Summary . . . 39

3 Threat Modelling in Collaborative Business Networks 40

3.1 Building Threat Modelling for Collaborative Business Networks . . . 41

3.2 Modelling of the Collaborative Network . . . 43
3.2.1 Defining the Participants and Their Roles . . . 43
3.2.2 Defining the Collaborations . . . 45
3.2.3 Defining the Assets . . . 46
3.3 Modelling the Attack and Threat Landscape . . . 47
3.3.1 Defining the Threat Actors . . . 47
3.3.2 Defining the Threats . . . 48
3.3.3 Mapping Actors, Assets and Threats . . . 51
3.3.4 Deriving Security and Privacy Requirements . . . 56
3.4 Addressing the Identified Threats . . . 58
3.4.1 Mitigating Threats . . . 58
3.4.2 Accepting Threats . . . 60
3.5 Validating the Threat Model . . . 60
3.5.1 Validating the Mitigations . . . 60
3.6 Threat Model of the Running Example . . . 60
3.6.1 Modelling the Collaborative Network . . . 61
3.6.2 Modelling the Attack and Threat Landscape . . . 63
3.7 Summary . . . 69

4 Scalable Privacy-Preserving Model for Collaborative Business Networks 70
4.1 Confidentiality-Enhanced Collaborations in Business Networks . . . 70
4.1.1 Hierarchical Grouping . . . 71
4.1.2 Creating a Secure Communication Infrastructure . . . 73
4.1.3 Integrating SMPC into Hierarchical Groups . . . 74
4.2 Confidentiality-Enhanced Lead Time Calculation of Wheel Production 76
4.3 Security and Privacy Analysis of Our Model . . . 79
4.3.1 Spoofing . . . 80
4.3.2 Tampering . . . 80
4.3.3 Information Disclosure . . . 81
4.3.4 Denial of Service . . . 82
4.3.5 Elevation of Privilege . . . 82
4.3.6 Linkability and Identifiability . . . 83
4.3.7 Non-repudiation . . . 84
4.3.8 Detectability . . . 84
4.3.9 Content Unawareness . . . 85
4.3.10 Policy and Consent Non-compliance . . . 85
4.4 Security and Privacy Analysis of the Running Example . . . 85
4.5 Summary . . . 86

5 Implementation of the Scalable Privacy-Preserving Model 87
5.1 Setting up System for Hierarchical Groups 87
5.2 Creating the Public Key Infrastructure (PKI) 88
5.3 Integrating Secure Multi-Party Computation (SMPC) into the Groups 93
5.4 Implementation Steps of Security- and Privacy-Enhanced Computation in a Hierarchical Group 97
5.5 Summary 100

6 Implementation of a Case Study: LCA 101
6.1 Life-Cycle Assessment (LCA) 101
6.1.1 Calculation of Traditional LCA 103
6.2 Modelling of Life-Cycle Assessment Network 109
6.2.1 Identifying the Participants and Their Roles 109
6.2.2 Identifying the Collaborations 110
6.2.3 Identifying the Assets 110
6.3 Modelling the Attack and Threat Landscape 111
6.3.1 Identifying the Threat Actors 111
6.3.2 Identifying the Threats and Mapping Them with Actors and Assets 111
6.3.3 Deriving Security and Privacy Requirements 117
6.4 Confidentiality-Enhanced Life-Cycle Assessment 118
6.4.1 Confidentiality-Enhanced LCA Model 118
6.4.2 Calculation of Enhanced LCA 122
6.5 Security and Privacy Analysis of Enhanced LCA Model 123
6.5.1 Mitigating Threats 124
6.5.2 Accepting Threats 127
6.6 Summary 127

7 Performance Evaluation 129
7.1 Experimental Setup 130
7.2 Traditional LCA: Time Complexity 130
7.3 Traditional LCA: Applying SMPC 136
7.4 Traditional LCA: Runtime Evaluation 137
7.5 Confidentiality-Enhanced LCA: Time Complexity 141
7.6 Confidentiality Enhanced LCA: Applying SMPC 144
7.7 Confidentiality Enhanced LCA: Runtime Evaluation 145
7.7.1 Selection of Benchmarking Scenarios 146
7.7.2 Evaluation of the Performance of the Confidentiality-Enhanced LCA Model through Experiments 148
7.7.3 Experimental Runtime Discussion 161
7.8 Summary 163

8 Related Work **164**
8.1 Addressing Security and Privacy Issues 164
8.1.1 Security and Privacy in Business Collaborations 164
8.1.2 Threat Modelling 165
8.1.3 Threat Modelling for Business Collaborations 167
8.2 Preserving Security and Privacy 168
8.2.1 Security and Privacy in Business Process Modelling 168
8.2.2 Privacy-Enhancing Technologies 169
8.2.3 Integration of Privacy-Enhancing Technologies in Business Processes 171
8.2.4 General-Purpose Compilers for SMPC 174

9 Conclusion and Future Work **177**
9.1 Conclusion 177
9.2 Future Work 178

List of Figures

1.1 The Relationships of the Technical Contributions of the Thesis 19

2.1 Setup of PKI System 27
2.2 Authentic Communication with PKI 27
2.3 Diagram of Secure Multi-Party Computation (adopted from [65]) . . 28
2.4 Calculation of an Average Salary in SMPC 33
2.5 Partial Bicycle Supply Chain and Participants Roles 36
2.6 Lead Time Computation in a Bicycle Supply Chain 37

3.1 Threat Modelling Process 42
3.2 Participant Roles in Business Networks 43
3.3 Asset Categories in Collaborative Business Networks 46
3.4 The Threat Actors of Business Networks 48
3.5 Bicycle Supply Chain Network Structure 61

4.1 Hierarchical Grouping of Running Example Supply Chain 72
4.2 Distribution of Certificates in Wheel Production Supply Network . . 73
4.3 Using SMPC for the Computation of the Wheel Production Supply Network 75
4.4 Secure Customer Order Lead Time of Wheel Production 78

5.1 Network Communication Architectures in a Hierarchical Group . . . 88
5.2 PKI for Communication Infrastructure 90
5.3 PKI for Computation Infrastructure 92
5.4 The Establishment of the Group PKI 93
5.5 Security- and Privacy-Enhanced Computation of One Hierarchical Group 97
5.6 Architectural Diagram of the Prototype Model for a Single Hierarchical Group 99

6.1 BPMN Diagram of LCA of Aluminium Production 104
6.2 LCA of Aluminium Production Supply Chain 105
6.3 Hierarchical Grouping of Aluminium Production Supply Chain 119
6.4 Distribution of Certificates in Aluminium Production Supply Chain . 120
6.5 Recursive LCA Computation for Aluminium Production Supply Chain 121

7.1 Flowchart of a Traditional LCA Using SMPC 137
7.2 Runtimes of the Traditional LCA with SMPC 138
7.3 Comparison of Matrix Inversion and Traditional LCA Performance in SMPC 141
7.4 Flowchart of LCA Computation for One Group in the Enhanced LCA Model 145
7.5 Structure of Supply Chain Scenarios 1–3 (From Left to Right) 146
7.6 Runtimes of Supply Chain Scenarios with Traditional LCA Calculation Method 149
7.7 Runtimes of a Flat Tree with Traditional LCA Calculation Method . 151
7.8 Runtimes of a Flat Tree with Optimized and Full LCA Calculation Methods 153
7.9 Runtimes of Supply Chain Scenarios with Optimized LCA Calculation Method 155
7.10 Runtimes of Supply Chain Scenarios with Optimized LCA Calculation Method 156
7.11 Runtimes of Supply Chain Scenarios using Different LCA Calculation Methods 159

List of Tables

3.1 Security and Privacy Properties and Their Corresponding Threats (merged from [101] and [103]) 49
3.2 Security and Privacy Threats and Mitigation Strategies 59
3.3 Mapping Threats to Threat Actors and Assets 68

4.1 Security and Privacy Threats and Corresponding Countermeasures . 79

7.1 Time Complexity of Operations in Traditional LCA 136
7.2 Direct Application of SMPC on LCA 138
7.3 Impact of Matrix Inversion on the Overhead of SMPC 140
7.4 List of Test Scenarios with Different Supply Chain Structures and Calculation Methods 148
7.5 Impact of Supply Chain Structure on LCA Computation Using SMPC Naively 150
7.6 Impact of the Number of Participating Companies on LCA Computation Using SMPC Naively 152
7.7 Impact of Operation Complexity on SMPC Overhead 154
7.8 Impact of Our Approach on the Performance of LCA Computation of Different Supply Chain Scenarios 158
7.9 Comparison of Traditional LCA Approach and Our Approach on the Performance of LCA Computation in Different Supply Chain Scenarios 161

8.1 A Summary of Defining Features and Documentation Types of Frameworks (mainly adopted from [250]) 175
8.2 A Summary of Functionality and Expressibility of Each High-Level Language (mainly adopted from [250]) 175

Acronyms

AI Artificial Intelligence.

BPMN Business Process Modeling Notation.

CA Certification Authority.

CN Common Name.

CSC Cyber Supply Chain.

DN Distinguished Name.

DP Differential Privacy.

FHE Fully Homomorphic Encryption.

FL Federated Learning.

GC Garbled Circuit.

HE Homomorphic Encryption.

IIoT Industrial Internet of Things.

IOI Items of Interest.

IoT Internet of Things.

ISP Internet Service Provider.

IT Information Technology.

LCA Life-Cycle Assessment.

LCI Life-Cycle Inventory.

LSSS Linear Secret Sharing Scheme.

ML Machine Learning.

MSP Monotone Span Programme.

OLT Order Lead Time.

OT Oblivious Transfer.

PHE Partially Homomorphic Encryption.

PKI Public Key Infrastructure.

SCM Supply Chain Management.

SMPC Secure Multi-Party Computation.

SWHE Somewhat Homomorphic Encryption.

VPN Virtual Private Network.

1. Introduction

1.1 Motivation

A major trend in the modern economy is the implementation of multi-party collaborations, where participants share and exchange information, data, resources, and responsibilities to achieve their goals. In these collaborations, organizations are shifting away from the traditional "closed" innovation paradigm, in which they rely on their own internal knowledge resources, skills, and production facilities for innovation design and implementation, towards an "open" innovation paradigm that relies heavily on collaboration with other organizations. This new paradigm involves leveraging external knowledge resources, skills, and production facilities to drive innovation [1]. In addition to transferring existing information between organizations, inter-organizational collaborations also facilitate the creation of new knowledge and the development of synergistic solutions [2].

In more detail, collaborations are important in the modern economy for several reasons [2, 3]:

- Increased efficiency: Collaborations allow organizations to pool resources and expertise, leading to more efficient and effective outcomes.
- Access to new markets: Collaborations provide access to new markets, customers, and distribution channels that may not have been accessible otherwise.
- Sharing of risks and costs: Collaborations allow organizations to share risks and costs associated with new projects and initiatives.
- Innovation: Collaborations encourage innovation and the development of new products, processes, and technologies.
- Competitive advantage: By working together, organizations can gain a competitive advantage over their rivals and stay ahead in an increasingly competitive global market.

The internet has greatly facilitated collaboration between organizations, allowing for the establishment of networks that support the execution of collaborative business processes. These networks consist of a group of diverse organizations, which may be autonomous and dispersed geographically, that are connected through the internet and work together towards a common goal or to perform a specific business process [4].

Smart manufacturing, also referred to as Industry 4.0 or the Fourth Industrial Revolution, is an innovative field that revolutionizes the way organizations collaborate by improving communication, collaboration, and knowledge sharing. While the Third Industrial Revolution, known as the Digital Revolution, facilitated collaboration through digital technologies, Industry 4.0 takes this to a new level by enabling collaboration across multiple domains and organizations. Industry 4.0 integrates advanced technologies such as the internet of things (IoT), artificial intelligence (AI), machine learning (ML), and cloud computing into manufacturing processes [5, 6]. This integration allows for more precise and real-time data exchange and analysis, leading to informed decisions and improved collaboration with partners, suppliers, customers, and even competitors. The adoption of Industry 4.0 transforms the design and implementation of collaborative business processes, such as supply chains, by connecting various processes to smart devices and advanced technologies. Data is shared in real time across multiple domains and organizations through these connected devices and advanced technologies [7, 8, 9, 10]. The integration of specialized and advanced technologies in Industry 4.0 enables organizations to work together more efficiently and effectively. By leveraging the benefits of IoT, AI, ML, and cloud computing, Industry 4.0 has the potential to revolutionize the manufacturing landscape and drive innovation in various industries.

The use of cloud computing has become a popular solution for sharing data and storing a large amount of data in business collaborations. This technology provides a platform that enables multiple organizations to access and share data, thereby enhancing collaboration and information exchange. With cloud-based solutions, organizations can collaborate in real time on the same data, resulting in more efficient and informed decision-making [11]. Cloud computing also offers increased flexibility and opportunities for remote collaboration, as collaborators can access data and systems from any location and on any device [12]. In addition, cloud solutions can reduce costs associated with setting up and maintaining IT infrastructure, allowing organizations to focus on their core business activities and improving collaboration with partners and customers [11]. The transfer of data to the cloud for collaboration purposes can lead to a change in the responsibility for ensuring data security and confidentiality: instead of being solely under the control of the organization, the responsibility falls to the cloud provider. The cloud provider can be either a trusted third party, not directly involved in the collaboration, or one of the participants. Regardless, the cloud provider is responsible for securely managing and processing the shared data. In this thesis, we provided an example for each cloud provider type in Section 2.4 and Section 6.1, respectively.

The current state of the art for inter-organizational collaborations requires data sharing; either this occurs directly amongst the participants or all participants need to trust the same third party. However, as data is often sensitive, organizations are hesitant to share their data with a third party or other organizations in the collaboration, due to security concerns [13] and strict confidentiality requirements [14, 15]. Additionally, certain collaborations may require participants to share data from sensors to make the computations more precise and capture real-time changes within these collaborations such as automated production processes [16, 17]. Nevertheless, this aspect also raises significant confidentiality concerns. Consequently, these concerns result in a reluctance to participate in close collaborations and hinder the formation of close relationships [15, 18, 19]. A solution that allows for secure and confidential computation without the need for sharing data in "plaintext" would solve this problem. In theory, secure multi-party computation (SMPC) [20, 21] provides a solution as it supports joint computations over confidential data while keeping this data private. SMPC also enables secure and confidential inter-organizational collaborations without the need for a trusted third party. Nevertheless, it comes with a significant cost: applying SMPC naively causes a severe performance penalty [22, 23, 24, 25]. Hence, it is not an easy, off-the-shelf solution that can just be "dropped in" to secure a system landscape supporting inter-organizational collaborations; it needs to be adapted to each use case. In this thesis, we propose a scalable privacy-preserving model to address this problem. Our model enables the development of a collaboration model that allows organizations to achieve their business objectives while minimizing the risk to the security and confidentiality of shared data. Moreover, it ensures acceptable performance levels, facilitating real-time results crucial for time-sensitive business collaborations. One instance where this becomes crucial is in demand forecasting. Accurately estimating future demand based on historical data and market trends often requires analyzing data in hourly intervals to capture short-term fluctuations [26]. Another illustration involves supply chain simulation: employing simulation models to evaluate various scenarios and strategies, often demanding computation over specific time intervals (e.g., minutes or hours, as shown in [27]). Our model presents a hierarchical grouping approach that reduces the risk of sharing unnecessary data with a large group of companies by working with smaller groups. This hierarchical grouping approach also enhances the efficiency of computations.

Establishing a secure and confidential collaboration model is crucial but not sufficent to attract organizations to participate in a collaboration. To effectively encourage organizations to join a collaboration, it is necessary to examine and resolve the security and privacy concerns that come with sharing data in collaboration. This can be achieved by conducting comprehensive security and privacy assessments of the collaboration, thereby addressing the security and privacy concerns organizations may have about participating. A comprehensive security and privacy assessment of collaborations can be achieved through the use of threat modelling, which is one of the recommended methods. However, its application in collaborative business networks is limited and often specific to certain scenarios or software-oriented objectives [28, 29, 30]. In this thesis, we develop a novel threat modelling approach to assist organizations in assessing and managing their risk of joining such inter-organizational collaborations. This approach offers a systematic method for creating mitigation strategies, as well as for designing and establishing secure and privacy-sensitive business collaborations within a network of organizations.

1.2 Contributions

The main focus of this thesis is the development of a scalable privacy-preserving model for inter-organizational collaborations. All the contributions made within this thesis are aligned toward achieving this goal. Our thesis has four main contributions. First, we propose a new scalable privacy-preserving model enabling joint computations within inter-organizational collaborations that protects the confidentiality of the shared data while achieving collaboration goals. This contribution addresses the question of how organizations can protect business-critical data during collaborations. Second, we present a threat modelling approach tailored to inter-organizational collaboration scenarios. The aim of this contribution is to help organizations understand the risks associated with data sharing within collaborations and to encourage their participation in such collaborations. Subsequently, to evaluate the applicability of our proposed scalable privacy-preserving model and threat modelling approach, we apply them to a specific domain called life-cycle assessment (LCA). We do a threat modelling of the LCA scenario and get the security and privacy requirements that we need to consider when instantiating our scalable privacy-preserving model to LCA. We then develop a confidentiality-enhanced LCA model, building upon the foundation of our scalable privacy-preserving model. Finally, we develop a prototype to demonstrate the practical implementation of our model, showcasing how it applies SMPC to inter-organizational collaborations. Additionally, we assess the performance of our approach across various LCA scenarios using the prototype. The first two contributions, the scalable privacy-preserving model and the threat modelling approach, are the conceptual contributions of our work. The other two contributions, the confidentiality-enhanced LCA and the prototype implementation, are the case study and the evaluation of our work. The essential relationships of the contributions of this thesis are illustrated in Figure 1.1 below. In more detail, as our contributions, we

1. propose a new scalable privacy-preserving model for inter-organizational collaborations that ensures the confidentiality of the data of the participants of the collaborations without causing a significant performance penalty, based on a combination of decomposing the necessary computations in collaborations and the use of secure multi-party computation (SMPC);

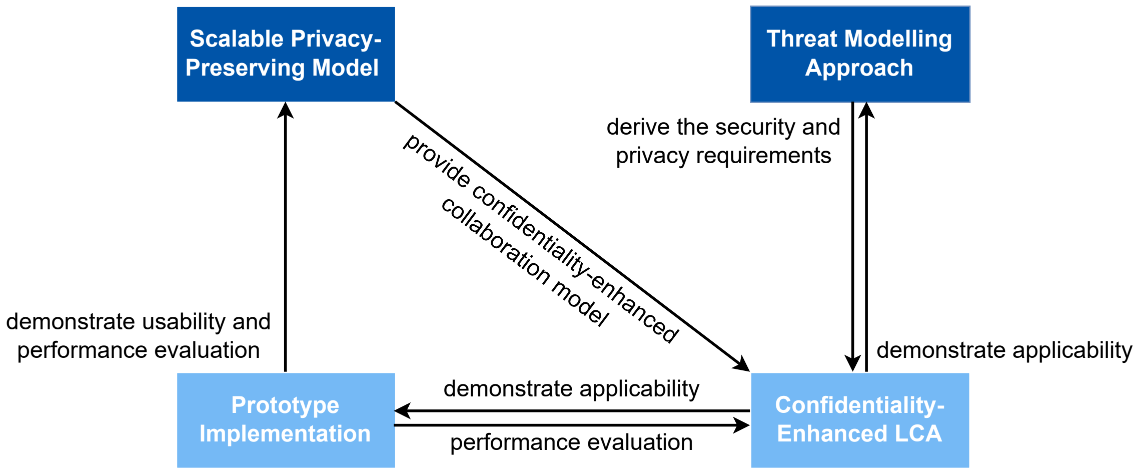

Scalable Privacy-Preserving Model
Threat Modelling Approach
derive the security and privacy requirements
provide confidentiality-enhanced collaboration model
demonstrate usability and performance evaluation
demonstrate applicability
Prototype Implementation
demonstrate applicability
performance evaluation
Confidentiality-Enhanced LCA

Figure 1.1: The Relationships of the Technical Contributions of the Thesis

2. develop a threat modelling approach for inter-organizational collaborations that allows companies to identify the threats associated with joining such collaborations and provides systematic guidance to developing mitigation strategies, and designing and developing secure and privacy-friendly collaborations in business networks;

3. develop a confidentiality-enhanced life-cycle assessment (LCA) model based on our scalable privacy-preserving model that enhances the confidentiality of the data of the participants of the supply chain, based on a combination of decomposing the LCA computation and the use of SMPC;

4. build a prototype based on our scalable privacy-preserving model demonstrating the use of SMPC for inter-organizational collaborations and evaluating the performance of our approach using LCA scenarios.

1.3 Thesis Structure

The remainder of the thesis is organized as follows:

- In Chapter 2, we give a summary of the background information related to collaborative business networks, with a focus on supply chain networks as an example. We provide a brief overview of supply chain management, including its objectives and the benefits and challenges of information sharing within supply chains. Additionally, we introduce the methods used in our work: public key infrastructure (PKI) and secure multi-party computation (SMPC), as well as the core services they provide. The chapter also includes an example of a simple collaboration within a supply chain, which is used as a running example throughout the thesis to illustrate the key aspects and concepts of the whole approach.

- In Chapter 3, we introduce a high-level threat modelling approach tailored to inter-organizational collaboration scenarios. The approach enables companies to identify the threats associated with joining a business collaboration and provides a process for determining the security and privacy requirements of a collaboration based on these identified threats. The chapter also offers systematic guidance on developing strategies to mitigate identified threats and on designing secure and privacy-friendly collaboration systems.

- In Chapter 4, we present a scalable privacy-preserving model for inter-organizational collaborations that enables confidential computation on shared data within collaboration networks. This chapter details the development of a model that reduces the amount of confidential data required to be shared and ensures its protection during the collaboration process. This chapter also demonstrates how to address the security and privacy threats identified in Chapter 3 using this approach.

- Chapter 5 builds on the material presented in Chapter 4, delving into the implementation details of our model. The chapter provides detailed information about the components needed to implement the model and discusses the technical decisions made to fulfill the requirements outlined in Chapter 3.

- In Chapter 6, we present a confidentiality-enhanced life-cycle assessment (LCA) model that has been developed using the approaches described in previous chapters. This chapter also provides a comprehensive assessment of the enhanced LCA in terms of security and privacy.

- In Chapter 7, we conduct a comprehensive performance evaluation of our scalable privacy-preserving approach, which was given in Chapter 4, in the context of LCA.

- In Chapter 8, we present a review of previous research that is relevant to our work and compare and contrast it with our own contributions.

- In Chapter 9, we provide the conclusion to the thesis and identify potential directions for future research.

Some of the text and research presented in various chapters of this thesis was previously published as part of the *Lecture Notes in Business Information Processing* book series. This published research paper [31] introduced our approach to building a confidentiality-enhanced business network model in the context of life-cycle assessment (LCA).

2. Background

In this chapter, we first give a brief overview of collaborative business networks, focusing on supply chains as a specific domain example. We present a brief summary of supply chain management, outlining its objectives, as well as the benefits and challenges associated with information sharing in such collaborations. Then, we introduce the methods used in our study; public key infrastructure (PKI) and secure multi-party computation (SMPC), along with the core services they offer. Finally, we introduce a simple supply chain collaboration scenario that serves as a consistent example throughout the thesis, aiding in the explanation of key aspects and concepts of the whole approach proposed in this thesis.

2.1 Collaborative Business Networks

In today's increasingly competitive business environment – including globalization, strong competition, the continuous emergence of new technologies, and the high expectations and demands of consumers, among other factors – individual companies find it challenging to design and implement the continuous stream of innovations essential for survival on their own. This motivates companies to form different types of business collaboration structures with other organizations, like collaborative networks, to design and implement product, service, and process innovations [1].

Depending on the context, the term "collaborative network" can have several meanings. It is frequently used to describe any type of network with some form of interaction, ranging from virtual professional groups to supply chains [32]. A more strict definition is proposed in our research work: a "*collaborative business network*" is a collaboration type where a set of companies shares their resources and performs tasks together in order to achieve their common goals.

The aim of the thesis is to offer a method for securing collaborations involving confidential data in business networks. Therefore, in order to provide detailed information about a collaborative business network and the collaborations that companies have within the network, we need to specify a collaborative business network scenario. In the following subsections, we will focus on a specific collaborative business network – the supply chain.

2.1.1 Supply Chain Management

The problem with the term *supply chain* is that it has nearly unlimited definitions. The majority of these are not in conflict with one another but opt to focus on certain aspects or features [33]. Christopher [34] gives a rather broad definition of a supply chain, defining it as a network of organizations that are involved, through upstream and downstream linkages, in the different processes and activities that produce value in the form of products and services in the hands of the ultimate consumer. An *upstream linkage* can be defined as supplying sources, whereas a *downstream linkage* might be a distribution channel [33]. The organizations in a supply chain may include various kinds of entities, such as manufacturers, suppliers, retailers, distributors, consumers, and the ultimate customers. These entities are linked by order fulfillment process(es) and the corresponding material, financial, and information flows to meet the needs of the end customers. *Supply chain management* (SCM) is increasingly being defined as the management of key business processes/operations throughout the network of organizations comprising the supply chain [35].

There are certain objectives to be achieved in SCM. SCM aims to deliver better customer service (efficient and effective services) while reducing overall costs and resources involved in the creation of products [36, 37, 38]. In addition to this, SCM also focuses on increasing profits [39, 40] and improving collaborations [41].

To achieve these objectives, it is important to encourage organizations to join supply chain collaborations and to share the information needed for these collaborations. In the following, we will provide an overview of information sharing in supply chains and discuss the benefits of and barriers to sharing information in a supply chain. The interested reader is referred to [42] for detailed documentation that explains SCM concepts, models, software, and case studies.

2.1.2 Information Sharing in Supply Chains

In order to survive in today's global economy, companies must constantly develop innovative methods of designing and delivering high-quality products and services in a timely manner [43]. However, as we stated previously, due to various factors such as globalization, intense competition, the constant emergence of new technologies, and high consumer demands, it has become difficult for organizations to come up with and execute a steady flow of innovative ideas on their own in today's highly competitive business environment. They need to rethink their approach to cooperation and within this need to find ways to share their up-to-date information with partners [44].

To thrive in today's economy, they are now networked to many other partners [44]. The networks of these collaborations form supply chains, and the collaborations in a supply chain require information sharing. A supply chain may contain many different types of information – such as logistic, business, strategic, and tactical [44] – and the type of information that needs to be shared depends on the collaborations that companies have in the supply chain. With recent developments in information technology (IT), the effect of information sharing on supply chains has become even more important. These advances in IT can help to produce new network structures that increase the closeness of coordination among supply chain partners. These structures can lead to more beneficial and profitable supply chains that may increase information flows, reduce uncertainty, and produce higher quality products with lower costs in a shorter period of time [44].

Benefits of Information Sharing

The overall aim of sharing information is to improve supply chain efficiency [43]. Sharing information among supply chain partners may provide a number of advantages to industries. Sharing information significantly contributes to reducing supply chain costs [45]; improving partner relationships [46]; increasing material flow [45]; enabling faster delivery [47]; enhancing order fulfillment rate, hence increasing customer satisfaction [48]; enhancing channel coordination [49]; and facilitating the achievement of competitive advantage [50]. Among these benefits, with the visibility of additional information (shared information), partners can improve their operational planning. For example, partners in a supply chain can make more accurate predictions with shared demand information [51, 52].

Although companies in a supply chain can (in principle) get perfect information about themselves, they might not obtain such perfect information about the other partners. Recent studies has focused on the advantages of information sharing for companies in the supply chain. However, more research is needed to determine how to share the benefits of information sharing between partners [43]. Lack of information can cause uncertainties for companies. To have efficient supply chains and reduce uncertainties, it is important for companies to be willing to participate in information-sharing activities. For more details, see [43, 44].

Challenges of Information Sharing in Supply Chains

Although sharing information within a supply chain has important benefits, it may encounter certain challenges [44]. These challenges can be listed as confidentiality of the information shared, incentive issues, reliability and cost of information technology, anti-trust regulations, the timeliness and accuracy of the shared information, and the development of capabilities allowing companies to utilize the shared information in an effective way [53, 54, 55].

Concerns about information confidentiality and privacy are one of the main barriers to information sharing. Most of the companies that would like to join supply chains are not willing to share their information with the other partners because of concerns about information confidentiality. This reluctance also acts as a barrier to the IT enablement of a supply chain [43, 55]. Lack of trust between partners within a supply chain may also impede information sharing [56]. In order to deal with these issues and encourage companies to share information, systems should generate a trusted network for partners and provide secure information-sharing platforms. However, the lack of formal frameworks and techniques that allow for confidential data sharing actually creates barriers to information sharing [43, 57].

Another challenge is that inter-organizational information systems are expensive to implement, time-consuming, and risky; in addition, there might be some goals that are not shared between partners. Moreover, there are other types of challenges and barriers to information sharing in supply chains: financial constraints, lack of commitment, lack of technology, and cost of technology [43].

2.2 Public Key Infrastructure (PKI)

A public key infrastructure (PKI) is a set of technologies, policies, and procedures that enable the deployment of security services based on public key cryptography. Public key cryptography is an encryption scheme using two mathematically related but not identical keys: a public key and a private key. A PKI is required for public key cryptography for one simple reason: to ensure that a particular public key belongs to a claimed entity [58]. In other words, a PKI is used to provide a trust model for public keys, establishing trusted public keys and distributing them. To enable a trust model, the PKI creates, stores, and distributes digital certificates. The certification authority (CA) of the PKI, which is an essential component of a PKI, first creates digital certificates in a way that binds public keys with related identities of entities using its digital signature, then sends them to the claimed entities. It also securely stores them in a central repository for use.

In the following, we first give a brief introduction to the PKI: the components of a simple PKI and the security services provided by a PKI. After that, we present an example that explains the implementation of a simple PKI. It is important to highlight that real-world PKIs are more complex: they provide additional services and have additional components. However, these are not discussed in this thesis; we only focus on the specific aspects of the PKI that are necessary for understanding the rest of the thesis. The interested reader is referred to the following for a detailed explanation: [59, 60, 61, 62].

2.2.1 Components of a PKI

A typical PKI consists of the following elements [59, 60, 61]:

- *A certification authority (CA)* acts as the root of trust in a PKI. It authenticates the identity of entities in a network by verifying and issuing their certificates; it confirms that the subject name imprinted on the certificate is the owner of the public key. Each certificate has the digital signature of the issuing CA. A signed certificate is called a digital certificate, and it contains the holder's public key and a number of attributes, which are the identified information about the holder of the corresponding private key.

- *Digital certificates*, also known as public key certificates, are used to cryptographically link the ownership of public keys with the corresponding entities that own those public keys. They are issued by trusted parties called certification authorities (CAs) that identify users or machines. Digital certificates are used to assure that the person sending the message is who they claim to be. A digital certificate contains at least the following information about the entity being certified: the certificate owner's public key, the certificate owner's *distinguished name* (DN) (which uniquely identifies the owner), the DN of the certificate issuer (which uniquely identifies the certificate authority), the issuer's signature, the dates for which the certificate is valid, and the serial number of the certificate [63]. When a digital certificate is presented to others, they can verify the identity of its owner with the help of the information included in the certificate. For example, it includes personal information like the owner's public key and distinguished name to identify and trace the owner. It also contains the issuing authority's distinguished name and its signature to identify and contact the issuing authority. These certificates are publicly known and may be securely stored in directories or databases.
- *Public/private key pairs* form the basis of a PKI for secure and authentic communications. This key pair functions in a unique way such that if a message is encrypted with one of the keys, the message can be decrypted only using the other key. One of the keys is kept secret, so it is called a private key, while the other key can be shared with anybody, which is referred to as a public key. Another important aspect of these keys is that they are mathematically related to each other, but it is computationally impossible to infer one of the keys from the other key. In a PKI, an entity first creates a public–private key pair, then sends a signing request including its public key and its identity to the CA. Then, the CA verifies the identity of the entity and binds the entity's public key and the identity of the entity using a digital signature (CA's private key). After that, the CA sends the generated digital certificate back to the owner.
- *A central directory* is a secure directory in which issued certificates are stored.
- *A certificate management system* manages services related to certificates, including accessing the stored certificates and delivering the certificates to be issued.

2.2.2 Core Security Service of PKI: Authentication

The core security service provided by PKI is authentication. The PKI authenticates that the key belongs to the entity named in the certificate. PKI provides authentication by verifying the identities of entities through digital certificates. A PKI includes a CA that is responsible for verifying the identities of entities and then binding their identities and their public keys with its digital signature (CA's private key) [59]. The entire concept of a PKI is built on trust. We trust that the CA issues certificates securely. If we do not trust the issuing CA, then we cannot trust any certificates that have been issued by the CA. A party that knows the public certificate of the CA and trusts the CA can easily verify the certificate of an entity by checking the digital signature and the identity (subject name) on the certificate. If it is signed by the CA and the identity of the certificate is the same identity that the entity claims, then the party can verify that the entity is who they claim to be. With an example, we will explain how the PKI establishes trusted public keys and ensures the authenticity of entities.

There are some limitations of PKIs that are directly relevant to our use of PKIs in this thesis: anonymity, trust dependency, and single point of failure. Firstly, a PKI requires identity verification through the use of digital certificates issued by trusted certificate authorities. The CN field contains the DN or IP address of the certificate holder for whom the certificate is issued. This linkage of real-world identities to cryptographic keys is a fundamental aspect of PKI's trust model, but it also compromises anonymity. In addition to this, PKI relies on the trustworthiness of certificate authorities (CAs). Users must trust that CAs are correctly verifying the identities of entities when issuing certificates. However, if a CA's practices or security measures are compromised, it can undermine the trust in the entire PKI system. Trust becomes a critical factor in PKI, and the loss of trust in one CA can have widespread implications. Finally, the centralized nature of PKI can create a single point of failure; the reliance on a single entity for issuing and managing certificates introduces a potential vulnerability.

For some of those limitations of PKI, we rely on contractual relationships that are usually established within collaboration scenarios, while for others, we discuss solutions. We will delve into this in detail in Section 5.2.

2.2.3 A Simple PKI Implementation Example

In this section, we present a simple example that explains how a simple public key infrastructure (PKI) implementation works. In our example, there are two people (Alice and Bob) who want to have authentic communication with each other through the use of a PKI. Figure 2.1 presents the setup of the PKI for this communication. In the example, we assume that there is a single certificate authority (CA) responsible for all services. At the start, each party gets the CA's public key certificate, and this is stored in their local data storage. Each party also has a key pair, which consists of the public key and the private key, held in their storage. To have a trusted public key certificate, Alice and Bob send signing requests for their public keys to the CA. The CA first verifies the identities of the parties and then binds each party's public key and its identity with a digital signature (CA's private key). After that, there is a distribution step where the CA sends back all signed certificates to each of the parties, Alice and Bob.

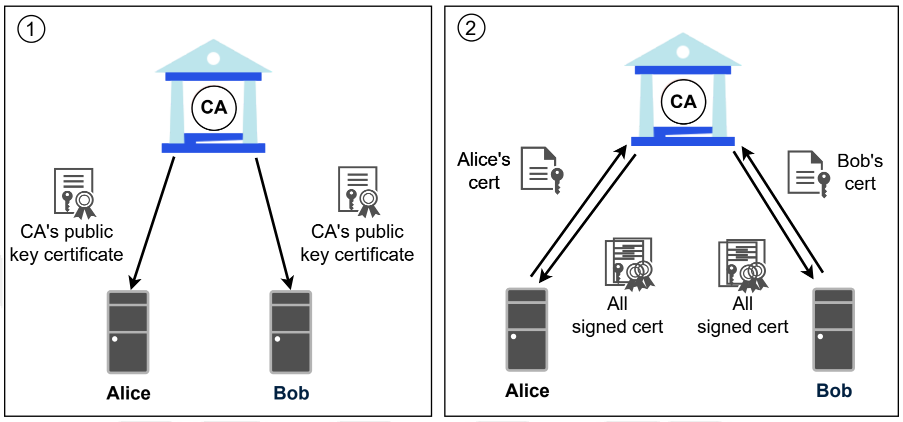


Figure 2.1: Setup of PKI System

When Alice communicates with Bob, Bob sends his digital certificate, which is signed by the CA, to Alice. Then, Alice uses the CA's public certificate to check the digital signature of Bob's certificate and the identity name on the certificate. If the signature is verified (belonging to the CA) and the identity name is the same as Bob claims, Alice knows that she is connected to the right person – the person whom they claim to be (in this case, Bob) (Figure 2.2). This is single-sided authentication. If we want to have mutual authentication in this communication, then the same steps should be taken for Bob too.

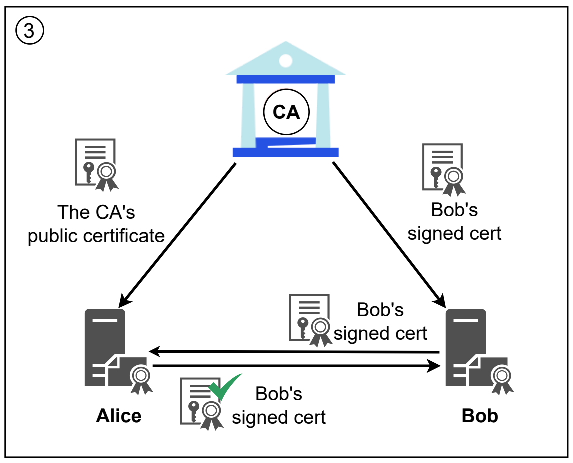


Figure 2.2: Authentic Communication with PKI

2.3 Secure Multi-Party Computation (SMPC)

Secure multi-party computation (SMPC) is a subfield of cryptography with the goal of creating methods that enable a set of parties to jointly compute a distributed arbitrary functionality on private inputs without revealing any confidential information but the result. Compared to traditional cryptographic tasks, where cryptography guarantees the integrity and security of communication/storage and the adversary is outside of the participants' system, secure multi-party computation is concerned with the protection of parties' privacy and confidentiality from each other in the computation [64].

In an SMPC setting, a given number of parties P_i $(i = 1, 2, ..., n)$ with private inputs x_i jointly and interactively compute the value of a public function on those private inputs: $f(x_1, x_2, ..., x_n) = (y_1, y_2, ..., y_n)$ while keeping their inputs secret. When the computation is completed, each party P_i should get its own corresponding output y_i without obtaining any further information [65]. This is illustrated in Figure 2.3. SMPC is aimed at building secure methods to allow distributed parties to compute this joint function over their inputs while preserving the privacy of the inputs and guaranteeing the correctness of the outputs even in the face of dishonest behaviour.

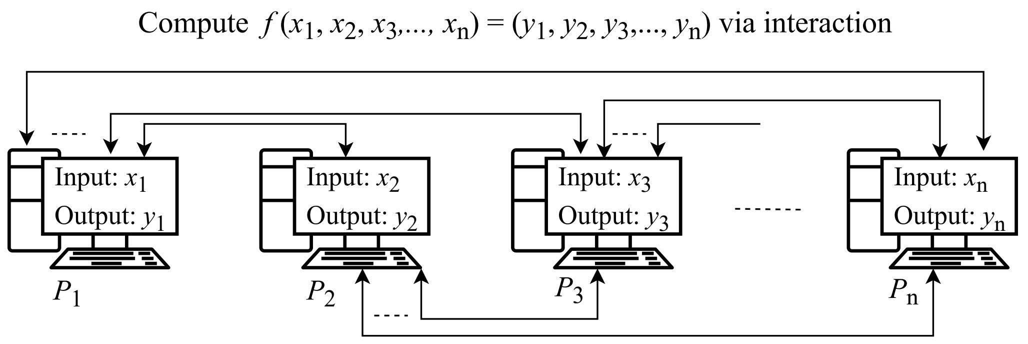


Figure 2.3: Diagram of Secure Multi-Party Computation (adopted from [65])

SMPC has been studied intensively in academia for over three decades and has strong theoretical foundations. In addition, huge progress has been made in the past decade to make SMPC efficient enough to use in practice [64]. Some use cases of SMPC that have been deployed include government collaboration [66], privacy-preserving analytics [67], SMPC for cryptographic key protection [68, 69], and advertising conversion [70]. Despite the progress of SMPC and many emerging real-world uses, challenges remain in applying SMPC to real-world problems, including cost, leakage trade-offs, output leakage, and meaningful trust [71]. These challenges must be overcome before SMPC can be deployed for a wide range of privacy-preserving applications.

2.3.1 Security of SMPC

Security requirements and definitional paradigm

SMPC considers the potential that parties in the computation task could behave dishonestly. A party as an attacker may aim to learn the private information of other parties in the computation or cause an error during the computation task. During the computation task, the attacker can take control of some subset of parties (called corrupted) and make them follow its instructions. To formally claim and prove that a protocol is secure against any adversarial attack, we should have a precise definition of security for multi-party computation. There are numerous definitions of security that have been proposed by researchers. They aim to have a number of important general security properties to cover most multi-party computation tasks. The most central of these properties are [64, 65]:

- *Privacy:* No party should learn any information beyond its prescribed output and what is intentionally shared during the execution of protocols. In particular, the only information that parties should be able to obtain about other parties' input is what they can derive from their own input or output.
- *Correctness:* Each party in a computation task should receive its correct output. This means the adversary must not be able to change the function that the parties set out to compute or cause the result of the computation to deviate from the correct result.
- *Independence of inputs:* The inputs of corrupted parties must be selected independently from the inputs of honest parties. It is important to note that independence of inputs is *not* covered by the privacy property.
- *Guarantee of output:* Corrupted parties should not be able to disrupt or prevent honest parties from receiving their own output. In other words, the computation should not be interrupted by any "denial of service" attacks that are carried out by the adversary.
- *Fairness:* The corrupted parties should be able to get their outputs if and only if the honest parties also obtain their outputs. The case where honest parties do not receive their output but corrupted parties do should not be allowed to occur.

The given list of properties above does not constitute a definition of security; they are the set of requirements that any secure protocol must satisfy. A definition of security, on the other hand, should cover all essential security needs as well as all possible adversarial attacks. It should also be simple and succinct enough for practical application [65].

In the current standard definition of security [72], an external trusted party is used to help individuals complete their computations in an *"ideal world"*. In this world, participants can send their inputs to the trusted party through a secure channel, and the trusted party will then perform the necessary computation and send the output to each participant privately. In this ideal world, the only opportunity for an adversary is to select the inputs of the corrupted parties, as the only action taken by participants is to send their inputs to the trusted party (since all other actions are performed by the trusted party). As a result, all security requirements listed above are met in this ideal world. However, in the *"real world"*, there is no external party that can be trusted by all participants. Instead, participants work together to execute a protocol without external assistance. If this protocol can produce the same results as the ideal world described earlier, meaning that no adversary can do more harm during the execution of the protocol in the real world than is possible in the ideal world, then the protocol is considered secure. This means that any successful attack carried out by an adversary in the real world must also be able to be successfully executed by an adversary in the ideal world, which is not possible. Therefore, all real-world attacks on the protocol's execution must fail.

More intuitively, we evaluate the security of a protocol based on comparison of two joint output distributions: that obtained by the adversary and the honest participants in the real world and that obtained through execution in the ideal world [65]. To determine if a protocol is secure, we consider how an adversary would perform when attacking the real protocol, and compare that to how the same adversary would perform when attacking the idealized version of the protocol (ideal world). If the input and output distributions (the combination of inputs and outputs) of the adversary and other participants are the same in both cases, then the security criterion is satisfied. When this criterion is met, it means that the real-world protocol reproduces the outcomes of the idealized version, or "simulates" the idealized version.

Security model

The informal definition of security given above excludes one very important issue, which is the capabilities of the adversary. The power of the adversary is defined by the security model of the system. Therefore, in order to have meaningful protocol security, the system needs to discuss the security of the protocol under a specific security model. The security models which categorize the power of the adversary can be divided into the following different types [65, 71, 73].

- *Semi-honest adversarial model:* In the semi-honest adversarial model, all parties, including corrupted parties, correctly follow the protocol as specified. However, as the adversary gets the internal status of each corrupted party, it can attempt to use this information to learn additional information that should remain confidential. Security in the presence of semi-honest adversaries ensures only a weak security guarantee: no inadvertent data leakage. These adversaries are also called "passive" or "honest-but-curious". Parties do not behave dishonestly or take any actions (like giving wrong inputs or interrupting the protocol/computation), and they trust each other. However, they, as an adversary, try to obtain as much private information as possible by observing the protocol execution.

- *Malicious adversarial model:* In a malicious adversarial model, corrupted parties arbitrarily deviate from the prescribed protocol according to the instructions of the adversary. A malicious adversary, also called "active", has the power of a semi-honest adversary, which is the analysis of the protocol execution, but can also behave dishonestly and take any actions it wants during the execution of the protocol. Certain protocols for achieving security in the presence of malicious adversaries can ensure that any adversarial attacks will fail. In this model, the privacy of honest parties is always preserved: the only thing that can happen in the case of a dishonest majority is to abort honest parties that have detected cheating. Although this model provides strong security guarantees, it requires high performance and causes inefficient computations.
- *Covert adversarial model:* A covert adversarial model is proposed to overcome the disadvantages of the other two models: the security of protocols in the presence of a semi-honest adversary is too weak, while the protocols based on a malicious adversarial model are too inefficient. The security in the covert adversarial model guarantees that if a covert adversary attempts to cheat, then it will be caught with some specified possibility that can be adjusted according to the application. This model contains more realistic situations where an adversary behaves maliciously only if they are not caught cheating. The covert adversarial model helps to find protocols that are efficient while being sufficiently secure for practical application.

When considering the definition and model of security to use, there is no "one-size-fits-all" approach. The specific definition and adversary that are chosen will depend on the specific application and the threats that need to be addressed.

Important definitional implications

There are important definitional implications that need to be explained [64, 74]:

- *The ideal/real paradigm used to define security has some significant consequences for the implementation of SMPC in practice.* To be more specific, companies that would like to use SMPC in their system need to examine their system security while an incorruptible trusted party performs the computation for which SMPC is used. If the system is secure in this instance, even when the real SMPC is used, the system will stay secure.
- Although the ideal model paradigm, as stated above, provides a straightforward abstraction, there are two subtle points that are sometimes misunderstood. First, *SMPC allows any input to be given.* SMPC behaves like an ideal execution; however, this does not mean that it prevents adversarial parties from inputting any values that they wish. There is no generic way to prevent this. Second,

SMPC secures the process, but not the output. Securing the process means that nothing is disclosed by the calculation. Nevertheless, this does not exclude the output of the function from revealing sensitive information.

2.3.2 Feasibility of SMPC

The concept of security described above looks to be quite restrictive in that no adversarial success is tolerated and the protocol should act as if the computation is being performed by a trusted third party. Powerful feasibility results have been obtained indicating that, in the presence of malicious adversaries, any distributed function (computing task) can be securely calculated [64]. The most important of these findings are summarized below. Let n represent the number of parties involved, and let t represent a limit on the number of parties that may be corrupted:

- For $t < n/3$ (i.e. when fewer than one-third of the parties involved in a secure multi-party protocol can be corrupted), it is possible to achieve secure multi-party protocols that ensure fairness and guaranteed output delivery for any function. This achievement relies on the assumption of computational security in a synchronous point-to-point network with authenticated channels [75]. Additionally, assuming the channels are private, information-theoretic security is also ensured [76, 77].
- For $t < n/2$ (i.e. if there is a guaranteed honest majority), secure multi-party protocols that ensure fairness and guarantee the delivery of output can be implemented for any function with both computational and information-theoretic security, given that the parties also have access to a broadcast channels [75, 78].
- For $t \geq n/2$ (i.e. when the number of corrupted parties is not restricted), it is possible to implement secure multi-party protocols, although fairness and guaranteed output delivery cannot be guaranteed in this scenario [75, 79].

To sum up, it is theoretically possible to develop secure multi-party protocols for any distributed computing task. This makes them a highly promising approach for computing tasks, as it allows for secure computation of any desired task. However, it is important to note that these feasibility results are based on theory and do not take into account the practical efficiency costs involved in implementing the protocols [64].

2.3.3 A Simple SMPC Example

There are various SMPC protocols used in SMPC approaches that help to provide computations under encryption; these protocols target different adversary models. There are several fundamental SMPC protocols [71] illustrating a variety of generic approaches to secure computation, and they have been developed against semi-honest adversaries, such as Yao's GC [80], GMW, BGW, BMR and GESS. On the other hand, some important SMPC protocols are designed to resist malicious adversaries [81]: BDOZ [82], SPDZ [83], MACE [84], SEPIOR, [85] and others [71].

In order to build SMPC protocols, a number of different techniques are used, such as secret sharing, homomorphic encryption (HE), oblivious transfer (OT), and garbled circuits (GCs) [65]. However, the most commonly used technique for constructing these SMPC protocols is the secret sharing scheme. Secret sharing schemes solve the problem of sharing a secret s among n parties by providing certain security. In secret sharing-based SMPC approaches, parties do not have special roles that take control of the computation. Instead, the secret is distributed between the parties in such a way that only when a sufficient number of the parties get together and combine their 'shares' can the secret be reconstructed. The shares of the secret are distributed to the parties based on the *threshold* of the secret sharing scheme. The threshold refers to the number of parties out of the group that must get together to reconstruct the secret. The interested reader is referred to [71] to learn more about SMPC, including fundamental SMPC protocols and implementation techniques. In the following, we illustrate a simple example that makes use of basic secret sharing to provide an idea of how SMPC works.

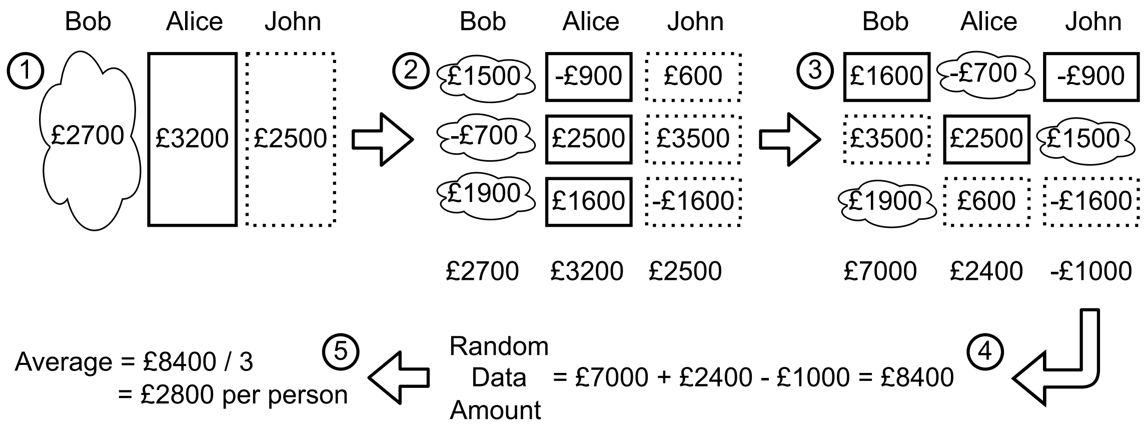


Figure 2.4: Calculation of an Average Salary in SMPC

Figure 2.4 shows our simple example for calculation of an average salary, where three colleagues would like to know the average salary since they all have the same occupation. However, they do not wish to share information about their own salaries. SMPC has a protocol using a form of secret sharing to achieve this in a secret way instead of using raw data. In this example, we consider salaries as secrets (1) and the threshold is the number of colleagues, which is 3. This means that to reconstruct the average salary, all colleagues need to get together. In the beginning, each employee splits their secret into shares (2). In this example, there are three shares for each secret, as we have three colleagues. Each share of the secret is assigned a random number, and the values of the shares sum up to the real value of the salary (secret). After receiving the shares of their secret, each colleague randomly distributes their shares among the colleagues. Thus, each colleague holds three random shares, and they get a new total of three (3). It is clear that the information held by each colleague is meaningless to them. In order to reveal the total salary of the three parties, they all need to get together. After this step, they can add their totals together and learn that the grand total is £8440, resulting in an average of £2800 per person (4 and 5).

As seen from this example, SMPC ensures the security requirements given in Section 2.3.1. With SMPC, the input data contributed by the participants is kept secret during the execution of the computation, and all participants only learn from the output (privacy property). The colleagues computed a joint calculation, and all received the correct outputs (correctness and guarantee of output properties). Not only output but also input information is important: the input of each colleague is selected on their own – a colleague does not know what the inputs of other colleagues are (independence of inputs). Last but not least, as they run a joint computation, in the case where honest colleagues do not receive their outputs, corrupted colleagues also do not receive theirs (fairness property). In conclusion, we can state that SMPC provides the creation of knowledge through a function without disclosing any secret data.

2.4 Running Example

In this section, we introduce an example of a simple collaboration within a bicycle supply chain. We will use this example in the following chapters of the thesis to illustrate the core aspects and concepts of our approach. In particular, we assume that the companies want to compute (and potentially optimize) the maximum customer order lead time, i.e. the time that a customer needs to wait between issuing an order and receiving their bicycle. In the following subsections, we first explain what lead time is in general and why it is important to determine. Then, we demonstrate how to calculate the maximum customer order lead time of a product with our simple bicycle supply chain example.

2.4.1 Determination of Lead Times

In today's competitive markets, most companies are not sufficiently skilled to design complete efficient production and distribution networks themselves. They are generally involved in multi-layer networks in which companies collaborate to transform raw materials into intermediate and final products that are then distributed to customers. A company gains a significant competitive advantage if it can design and then use a production and distribution network that meets the needs of its customers by providing high-quality products at low prices and with a short lead time. Therefore, decisions on how to design and configure such a network are an important and complex problem. These decisions can be very broad, from the selection of suppliers to the specification of flows of goods within the network. Although companies usually make such decisions to minimize costs, focusing only on cost minimization can lead to a serious breakdown in service quality. For this reason, introducing service-level constraints has become the preferred strategy in network design [86]. One of these service-quality constraints in the context of supply network design is lead time.

A *lead time* is the amount of time from the initiation to the completion of a process [87]. The definition of a lead time can be different depending on the domain it is applied to. Whereas in project management, lead time is the time required to carry out a task or a set of interdependent tasks, in the manufacturing sector it includes the time required to ship the parts to the supplier. In supply chain management, lead time is the time from receiving a customer order to it being ready for delivery [87].

There are some studies considering lead times in the planning context. Robinson *et al.* [88] developed a programming model to find the optimal supply chain for a manufacturer of power supplies. They model the time delays of material flows in multi-period systems. Bossert *et al.* [89] extended the guaranteed service supply chain modelling framework as investigated in so-called guaranteed service time approaches. When considering the measurements of service quality and the decisions made in network design, order lead time is a much less explored approach, and studies related to customer order lead times are very limited. However, some studies show that it is important and worthy of attention [86]. There are some works on the design and configuration of distribution networks [90], production plants [91], and a network of production stages [92] that consider the lead times of orders. Graves and Willems propose configuring a specific network of production stages for a single product by solving a selection problem in which customers' lead times are fixed through the selection of suppliers, parts, processes, and shipping modes [92].

2.4.2 Bicycle Supply Chain and Lead Time Computation

Now, we introduce a simple example that we use to illustrate our approaches in the chapters of the thesis. This example is a simple collaboration within a bicycle supply chain – in particular, the customer order lead time calculation. The supply chain illustrated in Figure 2.5 is a partial supply chain of bicycle production; it is simplified and does not include all processes. The supply chain of bicycle production presented in Figure 2.5 includes five raw materials, S1, CF1, R1, S2, and S3, four intermediate products, F1, W1, T1, and Sp1, and one final product, B1. We also have a final customer named as FC1 that obtains bicycle products from company B1. The supply chain also includes a trusted third-party called TP1. TP1 is responsible for calculating the customer order lead time of the bicycle production. The solid arrows represent the production flow, the consumer–supplier relationship. As can be seen from Figure 2.5, a company can be a consumer and a producer at the same time. For example, tyre company T1 gets rubber product from R1 company, but T1 company also produces tyres for wheel company W1.

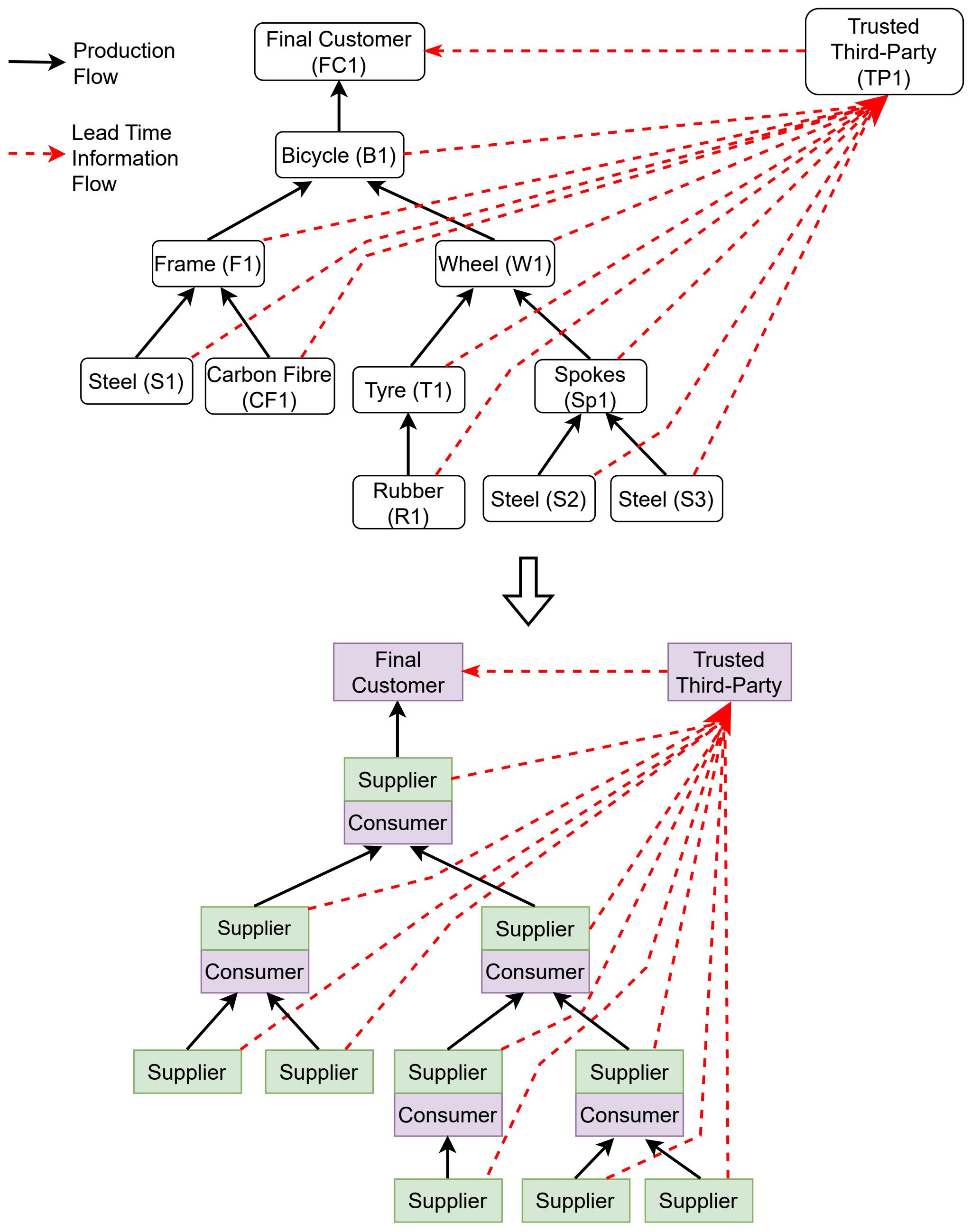


Figure 2.5: Partial Bicycle Supply Chain and Participants Roles

The dashed arrows represent the lead time information flow. All participating companies send the information to trusted third-party TP1 in order to get the customer order lead time result. In this supply chain, we determine the maximum customer order lead time of bicycle production through order processing times at the facilities (companies), including the times spent producing the raw materials, intermediate products, and final products, and the transportation times of the involved transport operations. The example is shown in Figure 2.6. The order processing time at a specified facility is labelled at the bottom of a vertex, and transportation times between the facilities are represented by the arc weights. For example, company F1 consumes steel (from S1) and carbon fibre (from CF1) materials to produce the frame of a bicycle. The lead time for the frame depends on the lead times for the steel and the carbon fibre (including transportation times) and the time required to produce the frame. In our example, company S1 (CF1) needs $t_{\text{prod}_{\text{steel}}} = 5$ days ($t_{\text{prod}_{\text{fibre}}} = 7$ days) to produce the steel (carbon fibre) and another $t_{\text{trans}_{\text{steel}}} = 4$ days ($t_{\text{trans}_{\text{fibre}}} = 3$ days) to deliver it to company F1. If we assume that company F1 needs 6 days for producing the frame and that the frame production can only start after all required materials have arrived at F1, the lead time of the frame can be computed as $\max(t_{\text{prod}_{\text{steel}}} + t_{\text{trans}_{\text{steel}}}, t_{\text{prod}_{\text{fibre}}} + t_{\text{trans}_{\text{fibre}}}) + t_{\text{prod}_{\text{frame}}} = \max(5+4, 7+3) + 6 = 16$ days.

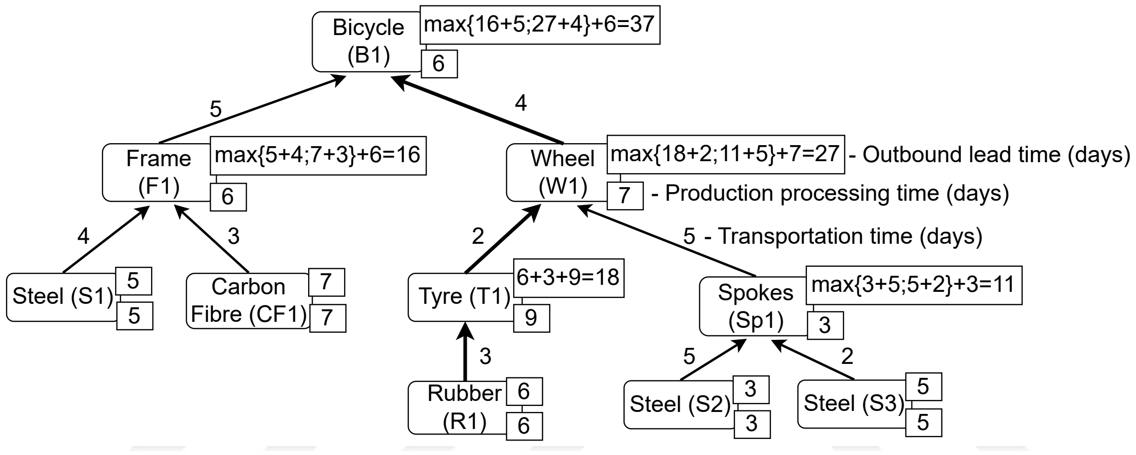


Figure 2.6: Lead Time Computation in a Bicycle Supply Chain

To calculate the maximum customer order lead time of the bicycle supply chain, we perform cumulative operations. Starting from the bottom of the tree of the supply chain (Figure 2.6), the outbound lead time is calculated in the group, and then the result is transferred upstream until reaching the final product, bicycle. In our example, we start with the latest downstream groups to make computations; we make the computations in parallel if they are independent of each other. One of the last downstream Order Lead Time (OLT) computations happens between Sp1, S2, and S3 companies. The consumer (Sp1) and its suppliers (S2 and S3) send their production processing times and transportation times to trusted third-party, TP1. After TP1 finds the maximum lead time among the suppliers of spokes company Sp1, which in this case is 8 days, it adds Sp1's production processing time of 3 days to obtain the outbound lead time of the spokes company, 11 days. In parallel, there are two other computations happening. Those computations are shown in Equation 2.1. One of them is between T1 and R1, and the other computation happens between F1, S1, and CF1.

$$\begin{aligned}
OLT(F1,S1,CF1) &= \max(t_{\text{outbound}_{\text{steel(S1)}}} + t_{\text{trans}_{\text{steel(S1)}}}, t_{\text{outbound}_{\text{carbon}}} + t_{\text{trans}_{\text{carbon}}}) + t_{\text{prod}_{\text{frame}}} \\
&= \max(5+4, 7+3) + 6 = 16\,\text{days} \\
OLT(T1,R1) &= t_{\text{prod}_{\text{rubber}}} + t_{\text{trans}_{\text{rubber}}} + t_{\text{prod}_{\text{tyre}}} \\
&= 6+3+9 = 18\,\text{days} \\
OLT(Sp1,S2,S3) &= \max(t_{\text{prod}_{\text{steel(S2)}}} + t_{\text{trans}_{\text{steel(S2)}}}, t_{\text{prod}_{\text{steel(S3)}}} + t_{\text{trans}_{\text{steel(S3)}}}) + t_{\text{prod}_{\text{spokes}}} \\
&= \max(3+5, 5+2) + 3 = 11\,\text{days}
\end{aligned} \tag{2.1}$$

After these computations, the next computation happens between W1, T1, and Sp1. Trusted third-party TP1 already has the production processing times of supplier companies T1 and Sp1, which are the results of the previous computations. The production processing times of T1 and Sp1 are actually their outbound lead times since they are intermediate suppliers. However, TP1 does not know the transportation times for tire and spokes products; it requests them from supplier companies T1 and Sp1. TP1 also requests the production processing time from consumer company W1. After TP1 finds the maximum lead time among the suppliers, in this case, which is 20 days, it adds W1's production processing time of 7 days into it to obtain W1's outbound lead time of 27 days (Equation 2.2).

$$\begin{aligned}
OLT(W1,T1,Sp1) &= \max(t_{\text{outbound}_{\text{tyre}}} + t_{\text{trans}_{\text{tyre}}}, t_{\text{outbound}_{\text{spokes}}} + t_{\text{trans}_{\text{spokes}}}) + t_{\text{prod}_{\text{wheel}}} \\
&= \max(18+2, 11+5) + 7 = 27\,\text{days}
\end{aligned} \tag{2.2}$$

The last computation happens between final consumer company B1 and its suppliers F1 and W1. The supplier companies send their transportation times to trusted third-party TP1. TP1 also requests bicycle company B1 to send its production processing time. After TP1 finds the maximum lead time between the suppliers, it adds B1's production time into it to obtain the maximum lead time of a consumer's order, which is 37 days, as given in Equation 2.3. As a result, the thick arches in Figure 2.6 show the way of getting the maximum customer order lead time in the bicycle supply chain.

$$\begin{aligned}
OLT(B1,F1,W1) &= \max(t_{\text{outbound}_{\text{frame}}} + t_{\text{trans}_{\text{frame}}}, t_{\text{outbound}_{\text{wheel}}} + t_{\text{trans}_{\text{wheel}}}) + t_{\text{prod}_{\text{bicycle}}} \\
&= \max(16+5, 27+4) + 6 = 37\,\text{days}
\end{aligned} \tag{2.3}$$

As we will see in the following chapters, already within this simple inter-organizational collaboration, there are risks in revealing confidential information to other participants in the collaboration.

2.5 Summary

In this chapter, we gave an overview of the background knowledge related to this thesis. We first provided brief background information on collaborative business networks, focusing on the supply chain domain as an example. We provided information about supply chain management along with its objectives, as well as the advantages and obstacles involved in information sharing within these collaborative networks. Additionally, we discussed the methods that we used in this study and highlighted their core services that we derived benefits from. Furthermore, we presented a simplified supply chain example that is used throughout the thesis to illustrate the key aspects and concepts of the overall approach in this thesis. It is important to comprehend the information presented in this chapter to gain a comprehensive understanding of the following chapters.

3. Threat Modelling in Collaborative Business Networks

In this chapter, we start to investigate how to build a systematic approach to assessing the security and privacy risks of collaborative business networks. Collaborative business networks and sharing information within such networks have indisputable benefits, such as allowing companies to improve their offer and competitiveness. However, due to increasing the interdependence of the participating components and the risk of revealing trade secrets, or increasing the attack surface for cyberattacks, companies are reluctant to enter close business networks [15, 93, 94]. Actually, the main factor that prevents close inter-organizational collaborations is fear of their impact on cybersecurity [93, 94, 95]. Securing collaborations within business networks, e.g. supply-chains, is not a new problem (see, e.g., [31, 96, 97, 98, 99]): a common suggestion is to use privacy-enhancing technologies such as homomorphic encryption (HE) [100] or secure multi-party computation (SMPC) [21]. As with most technologies for improving the security or privacy of business systems, they come with a significant cost, both during system development and operation. Hence, they are not an easy, off-the-shelf solution that can just be "dropped in" to secure a system landscape supporting inter-organizational collaborations.

Therefore, it is essential to conduct security and privacy analyses of such collaborative business networks to avoid the potential risks and the disclosure of any confidential information, and to ensure that the participants are protected, in order to encourage companies to participate in the networks. Also, it is important that companies can model and assess their individual risk of joining such an inter-organizational collaboration. The outcome of such a risk assessment can support companies in designing a collaborative business network that achieves their business purpose with minimal risk to the security and privacy of the shared data.

Threat modelling, e.g. as made popular by Shostack [101], is such a risk modelling and assessment technique that is widely used in software development. And while there are threat modelling approaches that support collaborations within business networks (e.g. [28, 29]), they focus on software development aspects. Their aim is to help to identify the security problems in software being built. However, to the best of our knowledge, there is still no threat modelling approach that has been developed with a focus on the business level. To fill this gap, we present a high-level threat modelling approach for flexible inter-organizational collaborations that focuses on the business level. The main goal of our approach is to help companies understand the risk of sharing data within a business collaboration and, therefore, to take appropriate countermeasures. In more detail, the contributions of this chapter can be summarized as follows:

- We develop a novel threat modelling approach for business collaborations that allows companies to identify the threats associated with joining such a collaboration.
- We define a process for deriving the security and privacy requirements of collaborative business networks from the identified threats.
- We provide systematic guidance to developing mitigation strategies and designing and developing secure and privacy-friendly collaborative business network systems.

3.1 Building Threat Modelling for Collaborative Business Networks

There are three structured approaches to threat modelling, which are centered on models of assets, models of attackers, or models of software [101]. As we are focused on the business level for inter-organizational collaborations, we conduct our threat modelling with a focus on assets. In this context, assets are the data being shared and used in the collaborations that attackers want to obtain and we want to protect. As distinct from traditional threat modelling approaches based on assets, we also focus on collaborations that can cause the disclosure of any confidential data. It is important to not only define assets and collaborations but also to determine attackers based on those assets and collaborations. Therefore, our threat modelling also considers the attackers in collaborative business networks. In the following sections, we describe how we combine these approaches and build our threat modelling approach. As stated above, there are other threat modelling approaches centered on software. Their focus is on the software being built or the systems being deployed. If we wished to carry out threat modelling focusing on software, we would need to do a more in-depth security analysis of the software we develop and deploy. However, this is beyond the scope of this thesis; we do not consider threat modelling that focuses on implementation or software issues in collaborative business networks.

The main aim of our threat modelling approach is to give companies systematic guidance on how to assess the risks of joining a collaborative business network and to develop mitigation strategies. In developing our approach, we follow the four main questions identified by Shostack [101], adapted to the domain of business collaborations:

1. What collaboration do we have?
2. What can go wrong?
3. How can we mitigate the identified problems?
4. Did we really identify and fix the important problems?

Starting from these questions, we designed our threat modelling process (see Figure 3.1), which comprises four main steps. Our aim is to define a generic threat modelling approach that captures a wide range of business networks and allows companies to identify threats associated with joining such networks. In the following sections, we start by defining the participants and their respective roles within collaborative networks, as well as outlining the nature of collaborations and the assets involved. This initial stage helps companies in effectively modelling their collaborative networks. Then, as the second step of our threat modelling process, we define the potential threat actors and threats and provide descriptive examples of how to map these threats, actors, and assets. Additionally, we list security and privacy requirements that participants in a business network might want to establish. This step is for modelling the attack and threat landscape within collaborative networks. The third step of our threat modelling process is to address the identified threats. We present some mitigation strategies used to reduce the likelihood of threats happening. Finally, in the fourth and last step, we define how to validate the threat model. After defining the whole concept, we present an example of how to apply our threat modelling approach. To represent the core aspects of our threat modelling in the following sections, we use our running example introduced in Section 2.4 as an illustrative case study.

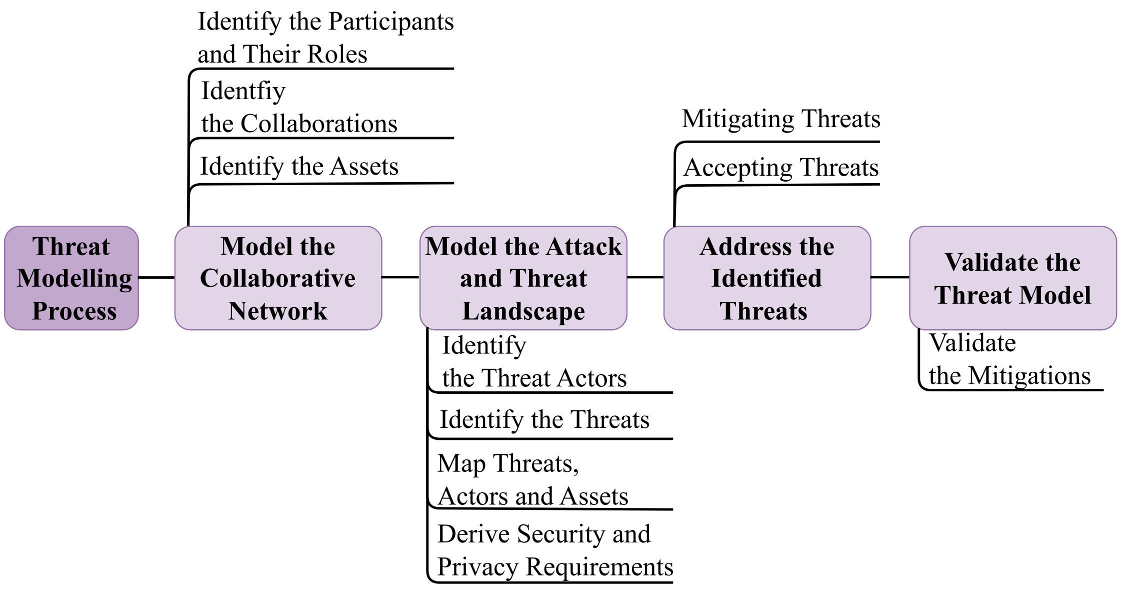

Identify the Participants and Their Roles
Identfiy the Collaborations
Identify the Assets
Mitigating Threats
Accepting Threats
Threat Modelling Process
Model the Collaborative Network
Model the Attack and Threat Landscape
Address the Identified Threats
Validate the Threat Model
Identify the Threat Actors
Identify the Threats
Map Threats, Actors and Assets
Derive Security and Privacy Requirements
Validate the Mitigations

Figure 3.1: Threat Modelling Process

3.2 Modelling of the Collaborative Network

The first step in building threat modelling of business collaborations is to model the network itself. We model the network by identifying the participants and their roles, the type of collaboration, and the assets (information) that need to be shared. In the following subsections, we define the likely participants and their roles, the type of collaboration, and the assets; we illustrate how to identify them through our running example.

3.2.1 Defining the Participants and Their Roles

In a business collaboration, participants (i.e. the actors) can play different roles. In general, we assume that the collaborating companies, either explicitly or implicitly, want to achieve a common goal. For example, in a supply chain, all participating companies want to deliver a service or a product to end-customers. In our threat modelling approach, we define various roles that participants can have within a business collaboration. This process helps participants accurately identify their positions within any collaboration. We have identified these roles based on the responsibilities of participants within a business collaboration (Figure 3.2):

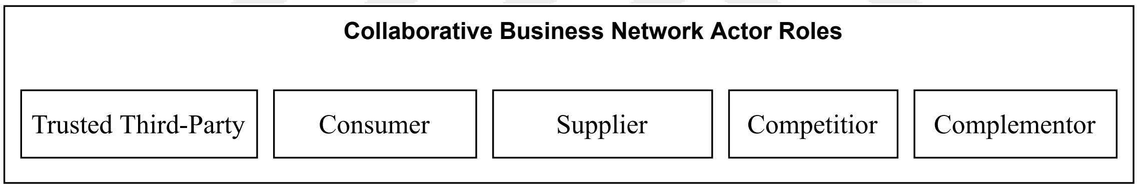


Figure 3.2: Participant Roles in Business Networks

- *Trusted third-party:* in many business collaborations, a trusted third-party is the company that acts as a bridge between the main actors in collaborations and is used to minimize the level of trust needed between them. For example, this can be a custodian managing the payment process to ensure that payment is only made after the goods have been received and checked by the customer. However, it could also be a service that provides computations over (potentially) confidential data of the main participants acting as a trusted third-party, such as a cloud server. Usually, the trusted third party has a contract (i.e. a legal business relationship) with one or several participants of the network. In our running example (Subsection 2.4.2), we have a trusted third-party called TP1 that provides computations over confidential data of the participants of the bicycle supply chain to calculate the order lead time of the supply chain.
- *Consumer:* a consumer is a company or end-user (client) that orders and consumes a service or product from a supplier. In our running example, frame company F1 is a consumer of the steel and carbon fibre products provided by

supplier companies S1 and CF1. Usually, there is a contractual relationship between a consumer and their direct suppliers.

- *Supplier:* a supplier provides a service or product to a consuming company, i.e. a company that uses the provided service or product to build their product (provide their service). In our running example, except for bicycle company B1, all companies provide their products to consuming companies to build their own products; therefore, they have a supplier role. Bicycle company B1 has a consumer role in this case. However, all companies, including bicycle company B1, also provide information (their production processing time and transportation time) to the trusted third-party TP1 in order for it to provide its service, which is calculating the order lead time of the bicycle supply chain. In this case, all companies have a supplier role.
- *Competitor:* companies that offer the same or very similar products or services often see themselves as competitors, i.e. as competing for the same customers. Hence, they are usually more reluctant to collaborate and, e.g., share data, compared to companies that offer different products. In our running example, the companies S1, S2, and S3 all produce steel and hence might see themselves as competitors. In particular, the situation where both S2 and S3 have the same consumer (Sp1) is critical from a security perspective.
- *Complementor:* complementors are companies that offer products and services that complement each other. Thus, by collaborating, complementors can provide a higher value to a mutual customer. In our running example, F1 and W1 could be understood as complementors: by joining forces, they could provide a frame with matching wheels to B1.

First, while we define the roles in a business network using the traditional flow of goods and services, our threat modelling approach focuses on (confidential) data and information exchanged. The reason is that we would like to identify the threats associated with joining such collaborative networks and sharing data in these networks. In our running example, for example, the data being transferred is the production processing time and transportation time of companies. Therefore, we focus on this data. Companies supply their production processing time and transportation time to trusted party TP1. In this case, they have a supplier role. Second, a company (actor) in a business collaboration can play several roles within the same or different networks. In the lead time supply network, tyre company T1 and spokes company Sp1 have supplier roles where they supply data to another company (trusted party TP1). However, they are also consumers that consume/use the product supplied by other companies in the network: R1 supplies their product to T1, and S2 and S3 supply their product to Sp1.

3.2.2 Defining the Collaborations

Business networks contain complex and systematic collaborations where companies have interdependent webs of relationships with each other. In general, we follow the general understanding (see, e.g., [102]) that business networks are temporary collaborations between companies that share a common goal. These collaborations can take multiple forms, each with its own characteristics, benefits, and purposes, including partnerships, open innovation, shared services, and non-profit and for-profit collaborations. Deciding which type of collaboration companies would like to have depends on what they want to get out of it. In our approach, we determine the type and scope of the business collaboration based on the roles of participants that they can play within a collaboration, leading to more generic forms of collaboration. We categorize these collaborations into three primary sub-groups as outlined below, while remaining open to extensions through specific instances or scenarios.

- *Co-opetition collaboration* is a collaboration type where companies are competitors. Co-opetition collaboration can reduce concerns for businesses by providing competitors with an opportunity to share their resources, prevent duplicating work, and generate new customers for all companies.
- *Portfolio collaboration* happens when a large company, which can act as a trusted third-party or a consumer company, takes on the responsibility of managing a broad collaboration with multiple smaller external companies or partners. These companies can play roles as suppliers, competitors, or complementors. The central managing company establishes and maintains the collaboration rules.
- *Network collaboration* is a collaboration where a group of companies works together with shared goals and values. This collaboration may or may not contain competitors. However, they have common interests that lead them to collaborate on mutually beneficial projects and share resources. The other two collaboration types we gave above are more customized; the network collaboration type encompasses them.

In our running example (Section 2.4), the common goal is to produce a bicycle. Usually, we want to limit the scope of our threat analysis. For example, within a supply chain, we might limit the scope of the analysis to a company and its direct suppliers. In our running example, starting from bicycle company B1, we could decide to consider frame company F1 and wheel company W1 as opaque entities, and, thus, the indirect suppliers (such as steel company S1 or tyre company T1) would not be in the scope of our analysis. When analysing the collaborations, it is also important to note the contractual relationships, as these can play a role in assessing the risk or help to identify information that, fundamentally, cannot be protected. For example, in most supply chains, a consumer will know the identity and financial details of its suppliers.

3.2.3 Defining the Assets

Next, we define the data (information) that needs to be protected. We use the term asset to describe anything that is of value to a company. Assets are usually scenario- or system-specific. Given our focus on holistic threat modelling for business collaborations, it is not possible to detail assets tailored to each individual actor or collaboration. Nonetheless, we can define them abstractly. During the asset identification process, we take into account the elements that hold significance in the collaborative context and require protection. In the following, we briefly discuss the main asset categories that we define in common business collaborations (Figure 3.3):

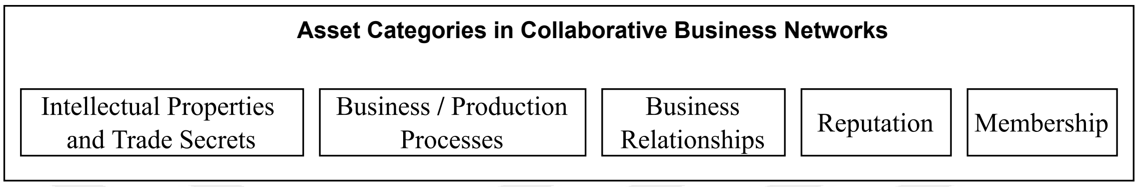

Asset Categories in Collaborative Business Networks
Intellectual Properties and Trade Secrets
Business / Production Processes
Business Relationships
Reputation
Membership

Figure 3.3: Asset Categories in Collaborative Business Networks

- *Intellectual property and trade secrets:* intellectual property for a business refers to legally protected assets that result from human creativity and innovation. A trade secret is a type of intellectual property that encompasses confidential and valuable business information not generally known to the public, giving a competitive advantage. Therefore, trade secrets and intellectual property of companies usually need to be protected within business networks. They might only be shared if other means of protection are in place, such as a signed non-disclosure agreement.
- *Business or production processes*: modern companies rely heavily on processes and their properties (e.g. efficiency, ability to work with certain raw materials) that can provide a competitive advantage. Thus, companies usually do not want to share the details of their business or production processes with other members of the business collaboration. In our running example, companies S1, S2, and S3 are competitors offering steel. If we assume that the lead time of a steel producer allows inferring, e.g., information about their internal production processes, they would not be willing to share the lead times with their competitors.
- *Business relationships:* often, companies consider their direct business relationships to be assets as well due to the benefits they bring, such as increased value, competitive advantage, sustainability, risk mitigation, and innovation potential. And they may keep them secret to maintain a competitive edge, protect sensitive information, and ensure exclusivity in partnerships. For example, a company might not want other companies to know about its suppliers. This could be individual pricing contracts negotiated or hiding their suppliers (e.g. competitors to the same consumer). In our running example, CF1 could exploit

the fact that it is the only supplier of carbon fibre to increase prices (and, hence, improve its profit). Thus, frame company F1 might have a legitimate interest that CF1 cannot learn that F1 does not have an alternative source for carbon fibre.

- *Reputation:* companies rely on their reputation (e.g. trust of customers); therefore, reputation is also a protection-worthy asset. Events that harm the reputation could, e.g., include the publication of customer data (e.g. a data leak), or information on collaborations with companies or political regimes that are seen as negative in certain communities.
- *Membership:* companies can be involved in more than one business collaboration or can play different roles within the same collaboration. This can, first, allow other participants of those networks to aggregate and combine information from different interactions. Second, sometimes the fact that a company participates in a certain collaboration (e.g. supplying material to a certain customer or nation-state) can be harmful, and therefore, companies might want to keep their participation information confidential (i.e. acting anonymously or pseudonymously). In our running example, knowing that both S2 and S3 companies offer steel to spokes company Sp1 could damage the relationship between Sp1 and its suppliers, S2 and S3.

These asset categories are not disjoint. For example, business processes can be understood as trade secrets as well. As the goal of threat modelling is to identify the assets that need to be protected, their assigned category is less relevant. Actually, it is beneficial to discuss the same asset from different perspectives to ensure that their value and protection worthiness are assessed correctly. To facilitate this, in real-world scenarios, our approach can seamlessly integrate with business modelling methodologies such as Business Impact Analysis (BIA). This methodology can help us to specify and prioritize assets within an organization in collaboration.

3.3 Modelling the Attack and Threat Landscape

After identifying the scope of business networks, we now turn to the threats and threat actors to examine what can go wrong with business collaborations. Then, we list the requirements that participants in a collaborative business network might want to establish. In this section, we discuss the step called *Model the Attack and Threat Landscape* from Figure 3.1.

3.3.1 Defining the Threat Actors

In collaborative business networks, there are two main types of threat actors: external and internal. An *external threat actor* is not a member of the business network. It attempts to exploit system vulnerabilities through the use of malicious software, hacking, sabotage, or social engineering. An *internal threat actor* is a member of the business network, i.e. someone who can act as a benign business partner in some transactions and, at the same time, also as a threat actor in others. While our threat modelling approach can also help to identify external threat actors, our focus is on internal threat actors. We distinguish attacks (see Figure 3.4) based on the relationship that threat actors and victims have: they can be *downstream, upstream, and lateral*, respectively. A *downstream threat actor*, for example, can be a consumer attacking its supplier, while an *upstream threat actor* can be a supplier attacking its customers/consumers. On the other hand, a *lateral threat actor* can be a company that might be targeting its competitors or other entities providing similar products or services to the same consumer. For example, in our running example (Section 2.4), steel company S2 could mount a lateral attack on steel company S3 to learn the pricing details of the contract between spokes company Sp1 and S3, and to outbid S3 in subsequent negotiations.

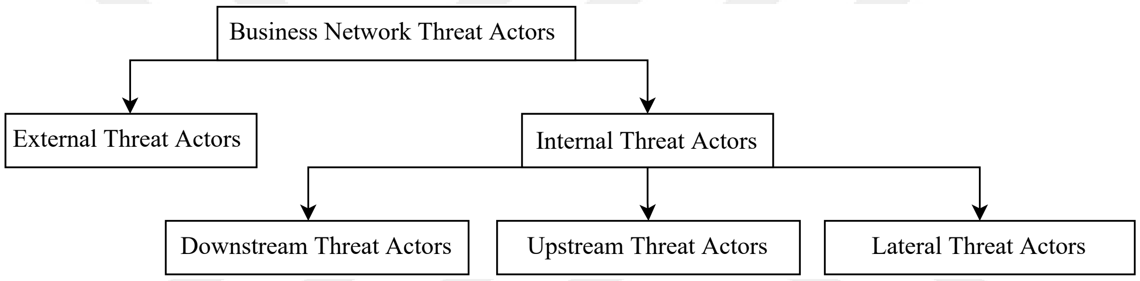


Figure 3.4: The Threat Actors of Business Networks

3.3.2 Defining the Threats

Once the threat actors have been defined, the next step involves identifying the threats that need to be considered within business collaborations. In the following, we discuss several threat categories that we believe are relevant to include in business collaborations. However, it is important to highlight that these are merely examples of potential threats. It is not possible to provide an exhaustive list of every possible threat scenario, as the threat landscape is constantly evolving and new threats emerge over time. For a more complete approach to covering threats, it is advisable to consider using multiple frameworks, methodologies, and tools that cover a wider range of threat vectors. To provide a comprehensive security and privacy assessment, we used STRIDE [101] and LINDDUN [103], which are widely accepted approaches to threat modelling [104], as a starting point for developing our threat categories. While STRIDE focuses on traditional security properties (e.g. spoofing, tampering), LINDDUN incorporates privacy-focused threat categories (e.g. linkability, identifiability). Therefore, we integrated these two approaches. In contrast to STRIDE and LINDDUN approaches, which primarily focus on eliciting and mitigating security and privacy threats within software architectures, our approach is centered on the business level, where the threats are tailored to business collaborations. Table 3.1 gives an overview of the security and privacy properties and their corresponding threats. In the following section, we will discuss them in more detail providing some examples.

Table 3.1: Security and Privacy Properties and Their Corresponding Threats (merged from [101] and [103])

Property	Threat
Authentication	Spoofing
Integrity	Tampering
Confidentiality	Information disclosure
Availability	Denial of service
Authorization	Elevation of privilege
Unlinkability	Linkability
Anonymity and pseudonymity	Identifiability
Plausible deniability	Non-repudiation
Undetectability and unobservability	Detectability
Content awareness	Content unawareness
Policy and consent compliance	Policy and consent non-compliance

As shown in Table 3.1, in our approach, we consider 11 types of security and privacy properties and their corresponding threats:

- *Authentication and Spoofing*: Authentication is a crucial process employed by businesses to verify the identities of entities attempting to access their systems or services. This property ensures that only authorized entities can gain access to their services, data, or networks, thereby protecting sensitive information and resources from unauthorized access. On the other hand, spoofing occurs when an attacker can successfully deceive a business system by falsifying one's identity or manipulating data to pretend to be a legitimate partner. As a result, it can gain unauthorized access to the business' services, data, or network.
- *Integrity and Tampering*: Integrity refers to the trustworthiness and accuracy of data, processes, and systems within a business collaboration. Maintaining data integrity is essential for making informed decisions and ensuring the reliability of business operations. While, an attacker, as an unauthorized actor, would like to modify (e.g. deletes or manipulates) data of businesses or business collaborations. This malicious act is commonly referred to as a tampering attack.
- *Confidentiality and Information Disclosure*: A business would like to have confidentiality property, which protects its sensitive information from unauthorized access or disclosure within a collaboration. Conversely, an attacker may attempt to share or release information with participants within or outside the collaboration, hence violating the confidentiality and privacy of the data.

- *Availability and Denial of Service*: A business would like to guarantee the uninterrupted availability of its services for legitimate customers. However, there exists a potential threat known as a denial of service (DoS) attack, in which an attacker might want to prevent the legitimate usage of a product or service, thereby obstructing its intended function.
- *Authorization and Elevation of Privilege*: In a collaboration, businesses would like to control access to specific resources, ensuring that only authorized participants can access them. This is achieved with authorization property, which involves granting or denying access rights based on the user's identity and assigned roles. Nevertheless, an attacker might want to gain authenticated access to another participant's systems and, through exploiting vulnerabilities in the software, manages to elevate their access privileges to a higher level. This potential attack is called the elevation of privilege.
- *Unlinkability and Linkability*: Unlinkability in a business collaboration refers to the ability to prevent actions or transactions from being easily connected to participants' identities, ensuring privacy and confidentiality. Linkability, on the other hand, describes the ability of a threat actor to link two or more items of interest (e.g. subjects, messages, actions, etc.) and, by that, infer information that should be protected.
- *Anonymity and Pseudonymity and Identifiability*: A business may desire to remain undisclosed and anonymous in a collaboration. In such cases, it can utilize pseudonyms or fake identities to participate in a business collaboration. Anonymity and pseudonymity enable this level of privacy. However, the anonymity or pseudonymity can be compromised if an attacker manages to associate an item of interest with the subject, thereby disclosing their true identity. This threat is known as the identifiability threat.
- *Plausible Deniability and Non-repudiation*: In a business collaboration, a business would like to have the ability to deny knowledge or involvement in certain activities. This refers to plausible deniability property. On the other hand, non-repudiation, as a threat, enables attackers to collect evidence that contradicts the assertions made by the party denying involvement, thereby establishing proof of a party's knowledge, actions, or statements. It is crucial to acknowledge that in certain collaboration scenarios, non-repudiation can be regarded as a fundamental security property. However, in our approach, we have chosen to categorize it as a privacy threat.
- *Undetectability and Unobservability and Detectability*: Undetectability and unobservability, in the context of business collaboration, refer to the ability of a company to avoid being detected or observed by competitors or other companies within the collaboration. However, there is a potential risk of detectability attacks specifically targeting the disclosure of confidential information, such as

a company's participation in a collaboration it intended to keep undisclosed or its traceability within the collaboration.

- *Content Awareness and Content Unawareness*: The content awareness property focuses on the company's consciousness regarding its own data shared during collaborations. The company would like to understand the consequences of sharing information. When companies are unaware of the content within the collaboration, not knowing the full context, they may unintentionally disclose excessive personally identifiable information or provide incomplete or inaccurate information. This can lead to compromised data security or hinder effective collaboration due to the absence of crucial context.
- *Policy and Consent Compliance and Policy and Consent Non-compliance*: Policy and consent compliance in business collaborations refers to the adherence to legal and ethical guidelines governing the sharing, use, and protection of information and resources between collaborating businesses. In a business collaboration setting, an attacker may breach agreed security or privacy policies, compliance regulations, or pertinent laws. Such violations can lead to consequences such as financial penalties or the loss of insurance coverage.

3.3.3 Mapping Actors, Assets and Threats

After actors, assets, and threats have been identified, it is important to create a map between them. The motivations and intentions of a threat actor may not be automatically known. As we do not have a specific scenario, we cannot make a detailed and specific mapping about which threat actor can attempt which attack on which asset we defined in previous sections. However, we can provide an abstract explanation and give some examples from our running example.

Spoofing

In collaborative business networks, if an attacker impersonates a participant, it can access and learn any information or modify any information that this participant has or could access. And this information can include private information (assets) of the participant. Collaborations between companies also affect the strength of the threat and the information (assets) that could be reached by the threat. Because companies collaborate on the network, the attacker has access to not only the information of the company that it spoofed but also some information of other companies with which this company collaborates. This information can also include different kinds of assets of those companies.

For example, in our running example in Section 2.4, due to a lack of authentication, steel company S3, as a lateral threat actor, might be able to convince frame company F1 that they are actually steel company S1. This could allow S3 to submit an over-priced quote to F1 that F1 believes to be from S1 and, hence, S3 might be able to take over the supplier relationship with F1. Another example is that if steel company S2, as an upstream threat actor, impersonates spokes company Sp1 (it has the consumer role at that level), it can learn about Sp1's business processes and relationships with S3, such as how much steel S3 provides and the production processing and transportation times of S3. Then, S2 can make some special offers to Sp1, like reducing the price or delivering the product quicker than S3, to undermine the relationship between Sp1 and S3 and become the only steel supplier to SP1 (gain profits). As seen from the example, having collaborative networks can cause the disclosure of more information than expected. S2 not only learns information about Sp1 but also about S3.

Tampering

In a collaborative business network, a threat actor can attempt to modify the data of a company that it uses and shares in the network. This can affect the assets of that individual company, such as reputation, intellectual process, the analyses of its production processes, and so on. At the same time, as this data is used in collaborative computations, it can affect the assets of other companies in the network. Tampering attacks can be conducted by different types of threat actors and can affect different asset categories we identified in our model.

For example, if steel company S1, acting as a potential lateral threat, were able to modify the data provided by steel company S3 to make its quality appear poor, it might be capable of damaging the reputation of S3 as a producer of high-quality steel. Not only the reputation of S3 but also the reputation of spokes company Sp1 can be destroyed since S3 is the supplier of Sp1. Or, if, as a downstream threat actor, the consumer Sp1 of the steel produced by S3 is able to modify the data of the ordering system of S3, it might be able to modify the price for a unit of steel, causing a financial loss to S3.

Information Disclosure

Some collaborative business networks can include a common platform so that everybody can communicate with each other. Also, the computation results of collaborations can be shared with all companies on those platforms. Although there is no intended disclosure there, if a company knows certain things in those computations or the results, it can easily reveal some information that it did not intend to share/publish. This revealed information can have an impact on different types of assets of companies and collaborations.

In our running example, companies might not want to reveal the production time to their competitors. Thus, they might only be willing to share their lead time with their direct customer, as making lead times public for different shipment options or destinations might allow a threat actor to infer the actual production time. Also, in our running example, if consumer F1 shares its outbound lead time with its suppliers, steel company S1 and carbon fibre company CF1, these two companies, as upstream threat actors, could potentially collude as upstream threat actors. This collusion might involve disclosing the production processing time of consumer company F1 within the supply chain, achieved by sharing their respective production lead times with each other.

Denial of Service

In collaborative business networks, a company can absorb the needed resources to prevent other companies from using the services. When companies cannot reach services, for example a service to send the data to the computation, this would affect the result of the computation. Although this attack has no direct effect on the assets, it would cause some problems, such as delays in processes (business/production and intellectual processes) and miscommunication between companies (business relationships).

In our running example, a denial of service attack on steel company S1, as a competitor, might force their consumers to switch to a different supplier (e.g. steel company S2) of the same material or service. Hence, in this example, S2, as a lateral threat actor, might be able to increase its profits by mounting a denial of service attack on the ordering system of S1. This not only destroys the business relationship between S1 and frame company F1, but also the business process of S1.

Elevation of Privilege

Elevation of privilege threats occur when a company gets rights or privileges that should not be available to them. In collaborative business networks, each actor has their own rights and privileges that are declared or specified by the system. If they gain a right or a privilege that they should not have, they can do or see something that they are not authorized to do or reach. Gaining specific rights or privileges can affect the assets of companies and the network.

For example, as a threat actor, steel supplier S1 might be interested in gaining access to information about other steel suppliers S2 and S3 (membership) that provide the same data/product to the network and their relationships (business relationships) in this network in order to learn about potential consumer companies like spokes company Sp1 and make a special deal with them.

Linkability

Linkability attacks in a collaborative business network can be conducted by all threat actors who can link two or more items of interest (subjects, messages, actions, etc.) within a network or networks. These links can reveal private information that affects all given assets.

For example, if a competitor company, as a lateral threat actor, learns that there is just one other competitive company that provides the same data type to the network computation, when the result of the computation is released, it can subtract its data and then link the rest of the result with the other competitive company. Another example could be that a complementary company would like to know the supplier company of its connector company in the network to make a special deal with that supplier company instead of the connector company. If the complementary company can link some items in the network, it can achieve its aim. In addition to this, if a participant has several roles within the same collaboration, it might be able to link data and infer new information. In our running example, let's assume that tyre company T1 and spokes company Sp1 are the same company, which supplies tyres and spokes. Then, the company supplying tyres and spokes might be able to infer if they are the sole supplier or if an alternative source, e.g. for the spokes, is used as well by wheel company W1.

Identifiability

Identifiability attacks aim for anonymity and pseudonymity properties in systems. In collaborative business networks, collaborations can include the subjects (companies) and their attributes (messages, actions, etc.), which can cause the disclosure of information affecting the assets of the companies given above (in Subsection 3.2.3). If a company identifies the message of another company in the network, it can reveal some confidential information from that message.

For example, a trusted third-party or consumer company in a collaboration can need to know that certain information comes from certain companies. In our running example, trusted third-party TP1 requires companies to send their production processing and transportation times to make required calculation. As TP1 can identify which company sends which data, this violates the anonymity and pseudonymity of companies.

Non-repudiation

Non-repudiation attacks in collaborative business networks can be made by all companies that can get evidence of other companies' actions or are responsible for providing, controlling, or managing the services in the networks, and they can affect all the assets we identified in Subsection 3.2.3.

In our running example, we might have the situation where wheel company W1 denies having received the tyre produced by T1. In this case, we want to have non-repudiation so that T1 can prove that they have delivered the tyre, and W1 has to pay T1. On the other hand, in a quote process, we might want to keep the names of the companies secret and even want to make sure that a company can plausibly deny having submitted a quote.

Detectability

Detectability is an attack where an attacker can detect the existence of items of interest (IOIs). In a collaborative business network, for example, a supplier (e.g. S1), as a lateral threat actor, can want to detect whether there are other competitive supplier companies (e.g. S2 and S3) in the network and to whom they supply their products (e.g.Sp1), in order to discover potential consumer companies. Not only suppliers, but other actors in the network can also be keen to know the existence of certain IOIs to use them for their profit. This detection can cause the disclosure of information that affects assets or directly impacts the companies in the network.

For example, frame company F1, as a downstream threat actor, can want to detect the potential steel suppliers (S2 and S3) in the network. If F1 convinces these suppliers to collaborate together (by offering a better price or conditions than spokes company Sp1), they can end their collaboration with Sp1, and this can cause the loss of participation of Sp1 in the network.

Content Unawareness

Content unawareness is related to information disclosure attacks. In a collaborative business network, if a company provides more information than required or inaccurate information to the network, it can cause problems in retrieving the identity of the company or of making incorrect decisions or taking incorrect actions.

For example, if steel competitors S2 and S3 provide their production processing time and transportation time separately instead of the lead time of the steel production to the consumer Sp1, Sp1 can use this information for its own profit, like choosing relationships with suppliers and changing transportation types. For example, S3 has a better lead time for steel production. Sp1 is aware, however, that the processing of steel production in S3 takes longer than in S2. Sp1 can offer S2 to change the transportation method to make the collaboration instead of collaborating with S3.

Policy and Consent Non-compliance

Depending on the business, there can be security and privacy requirements and policies required by law or industry standards, and these might require additional considerations.

In our running example, as there is a contractual relationship between a consumer and its direct suppliers, suppliers accept sharing some data with their consumers. However, if the consumer is in non-compliance with the promised policies and regulations, there can be risks in revealing the confidential data of its suppliers.

3.3.4 Deriving Security and Privacy Requirements

From a system design perspective, it is usually preferred to talk about requirements, i.e. properties the system should fulfill, as opposed to threats. Thus, prior to planning how the threats are mitigated, it is important to map the threats to security and privacy requirements that the collaborative business network needs to fulfill. From the threat landscape that we explained in the previous subsections, we derive the security and privacy requirements for business networks, including collaborations between companies. Although it may not be possible to achieve the complete security and privacy requirements that cover all business collaboration applications, as they can have specified system requirements, we list security and privacy requirements that participants in a collaborative business network might want to establish.

First, *authentication* is a security property of verifying that an individual, entity, or website is who it claims to be. In order to prevent any spoofing attack on the network, the identity of participants in a collaborative business network *should be authenticated*. On the other hand, although authentication is important, some business networks may want to treat their participating companies anonymously. The membership information of companies in a collaborative business network can be considered confidential due to the possibility of revealing some private information about companies – for example, how many networks they are participating in and their business relationships. Knowing this information can also affect the reputation of companies. Business networks should have *anonymity and pseudonymity* privacy properties to keep secret the identity and membership information of any participating company. By keeping participants' identities and actions anonymous, collusion also becomes challenging as it is difficult for colluding companies to identify each other and coordinate their efforts.

Another desirable privacy property is *unlinkability* property. A company in a collaborative business network may know how many companies are involved in the network. However, in order to achieve *unlinkability*, it should not know who they are and what their activities are if it does not have a direct relationship with them. This means that companies should not be aware of each other if they do not have a direct relationship (*unlinkability* property). For example, in our running example, wheel consumer company W1 would know the tyre company T1 and spokes company Sp1 working under its agreement, but it should not know any company/participant in the network that it does not have a direct relationship with, such as the membership information of sub-suppliers R1 (rubber), S2 (steel), and S3 (steel) of its contractual companies in the network. This property also makes it harder for colluding companies to identify each other's actions, preventing them from collaborating based on the linkage of their activities.

If a business network does not want any unauthorized companies to access any data that they are not authorized to see, then they should identify the rights and privileges of companies in the business network (*authorization* property). Also, *integrity* is the security property of preventing attempts to modify the data of a company that is used and shared in the network. To provide the *integrity* property, companies should not be able to modify any data that does not belong to them.

In some collaborative business networks, companies may know some data agreed to be shared in the network or in a computation of that network, but if they want to have the *confidentiality and content awareness* properties, this data should not reveal any (confidential) information that affects or discloses the assets (like business, production, or intellectual processes) of individual companies or the processes of collaborations in the network. Any data that has the potential to reveal or detect confidential data should be protected. For example, in the calculation of lead time in Figure 2.5, companies send their production processing and transportation times to TP1 to learn the maximum lead time in that supply chain. TP1 does not need to know the production processing and transportation times separately; it would like to know the maximum lead time for supplying products. Therefore, it should not learn these values but just the maximum value. Another example is where a company in a collaborative network may know some data, like the names of materials agreed to be shared in the computation, but it should not know the confidential data of other companies, such as which materials they provide to the network and the amount of material they provide.

Besides this, some companies (actors) in a business network, like trusted third-party or consumer companies, can specify the data being computed in the computations of collaborative business networks and get the data from other companies to carry out the computations. However, if participants would like to have the *undetectability* property, those companies should not be able to detect the existence of items of interest in those computations, such as the company and its inputs. This also ensures that the colluding participants cannot identify each other's inputs or collude to deduce extra information during the computation. Some companies and collaborations might require not only business, production, or intellectual process data but also information containing the business relationships of companies in collaborative business networks. This information might also cause detection threats. To achieve the *undetectability* property, this information also needs to be protected. For example, a consumer company should not know whether its suppliers have any supplier companies or who they are/how many there are. Another example could be that a competitor company should not know the relationships of its competitors.

In addition to these, the business objectives should be considered. Some collaborative business networks would like to have the *policy and consent compliance* property. If a system is affected by the regulations determined by law or industry standards, it is important to check those regulations and understand how they impact the assessment of the system. Policy and consent compliance also can help to prevent collusion; This can be achieved by creating clear regulations, enforcing strict monitoring and penalties, promoting transparency, fostering competitive environments, and obtaining informed consent from stakeholders.

As an ideal system, all companies/participants in a collaborative business network should be aware of the companies that they have direct relationships with and the collaborations in which they are participating. Beyond this, they should only receive the information that they need for analysing the market, evaluating their own profits, and having a more sustainable network/system.

3.4 Addressing the Identified Threats

While, ideally, we would like to have a system that only shares data on a need-to-know basis and allows participants to stay as anonymous as possible, this will usually not be possible. Furthermore, collaborative business networks do not operate in a "technological vacuum" but rather in a legal and societal environment that requires certain data to be shared or allows for data protection through legal and contractual means as an alternative to technical means.

Thus, threats can either be mitigated on a technical level or the associated risk might be accepted by a company. In both cases, this should be a conscious and fact-based business decision that considers the associated risk (i.e. likelihood and impact of a successful attack) and the impact (e.g. costs, delays) of implementing a mitigation strategy. In the following, we provide a brief explanation of mitigating threats and we give a very general list of mitigation strategies.

3.4.1 Mitigating Threats

Mitigating threats on the technical level requires selecting the appropriate security controls (e.g. access control, encryption) or privacy controls (e.g. privacy enhancing technologies such as secure multi-party computation) for data being processed, stored, or shared. Moreover, this also requires a secure architecture that minimizes the trusted computing base (i.e. minimizing the system elements that process or store critical data) and also ensures secure development practices (e.g. defensive programming) for the system itself. In addition (sometimes also as an alternative), threats might be mitigated on a legal or contractual level, e.g. by taking out insurance covering identified threats. In Table 3.2, we present some mitigation strategies used to reduce the likelihood of threats happening. It is important to note that the list we provided in the table is a very general list of mitigation strategies. For more information on threat mitigation strategies and techniques, we refer the reader to [101, 103, 105].

Table 3.2: Security and Privacy Threats and Mitigation Strategies

Threat	Mitigation Strategies
Spoofing	Authentication [106, 107, 108], Digital signatures [109], Secure communication (HTTPS, IPsec, IKE)
Tampering	Digital signatures [109], Authorization [110], Secure communication (HTTPS, IPsec, IKE)
Information disclosure	Encryption techniques [111, 112], Cryptographic protocols [20, 113], Access control techniques [114, 115, 116, 117]
Denial of service	Detection mechanisms (Signature-based detection [118], Anomaly-based detection [119]), Response mechanisms (Filtering or rate limiting [120], Capability-based response [121])
Elevation of privilege	Access control techniques [114, 115, 116, 117]
Linkability	Authentication [106, 107, 108, 122], Data anonymization [123], Information hiding [124, 125], Anonymity and Pseudonymity systems [105], Cryptographic protocols [20, 113]
Identifiability	Authentication [106, 107, 108, 122], Data anonymization [123], Information hiding [124, 125], Anonymity and Pseudonymity systems [105]
Non-repudiation	Deniable encryption [126], Deniable authentication [107, 122]
Detectability	Information hiding (Covert communication [124], Steganography [125] etc.), Anonymity system [127]
Content unawareness	Feedback tools for user privacy awareness [128, 129]
Policy and consent non-compliance	Policy enforcement and communication tools [130, 131]

3.4.2 Accepting Threats

Certain risks might be accepted by businesses because the risks are considered to be low (e.g. having a low likelihood of occurring and a low or medium impact), or the required mitigation is too costly in relation to the potential impact.

3.5 Validating the Threat Model

Any threat modelling or planning of mitigation strategies is only valuable if actions are taken. Hence, it is important to validate that the model covers all important aspects and that the mitigation strategies have been implemented effectively. In Chapter 6, we validate our threat modelling approach with an LCA case study.

3.5.1 Validating the Mitigations

When planning mitigation strategies, it is also necessary to plan validation and testing activities that ensure that the mitigation strategies are actually effective. For technical mitigation strategies, for example, this could include code reviews during the development of the system or penetration tests after the development of the system. For legal or contractual mitigation strategies, checkpoints should be defined to ensure that contracts such as cyber insurance are in place and active. Ideally, each mitigation strategy has an individual validation action associated with well-defined success criteria.

3.6 Threat Model of the Running Example

In this section, we do threat modelling of the lead time computation of bicycle production for one company based on our threat modelling approach. We would like to perform threat modelling from the perspective of one company. For this example, we used the *wheel company* from our running example in Figure 2.5. Before we begin threat modelling, we first need to establish the wheel company's assumptions; specifically, what the wheel company knows about the supply chain. Bicycle company B1, as the main consumer company in the supply chain, shares the following information with the wheel company: the business objective of the supply chain and the whole supply chain network depicted in Figure 3.5.

In this example, the business objective of the supply chain is to produce bicycle products and then deliver them to the final customer on time and in sufficient quantity. In order to achieve this objective, they compute the order lead time, which calculates the time from receiving a customer order to being ready for delivery. The calculation is performed by *TP1, a trusted third-party.* Participating companies give required inputs to TP1 for the order lead time computation of the supply chain:

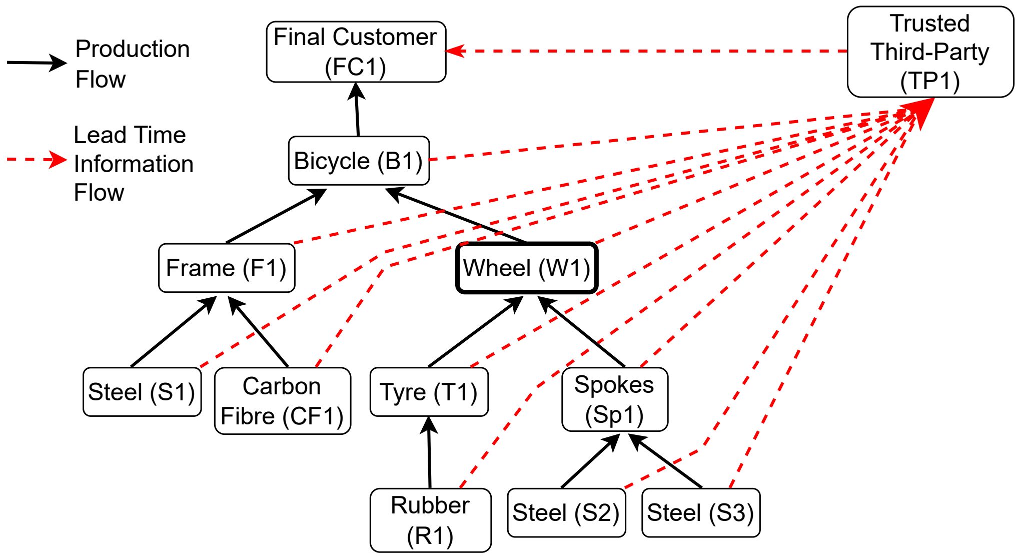


Figure 3.5: Bicycle Supply Chain Network Structure

production processing time and transportation time. Bicycle company B1 shares the supply chain network information described in Figure 3.5. The network information allows the wheel company to learn about the supply chain's participating companies and their relationships with each other.

We begin building threat modelling for wheel company W1 using our threat modelling approach in the following subsections. This section focuses solely on the collaborative network modelling and the attack and threat landscape modelling components of our threat modelling approach. We will focus on the other two threat modelling processes in Chapter 4, which are addressing the identified threats and validating the threat model. We will demonstrate how our scalable privacy-preserving model can mitigate these threats.

3.6.1 Modelling the Collaborative Network

The first stage in establishing threat modelling for the wheel company is to model its network in the bicycle supply chain network. To model the network of wheel company W1, we follow the steps in modelling the collaborative network stage of our threat modelling approach shown in Figure 3.1. These steps are, in order, identifying the participants and their roles, the types of collaboration, and the assets that need to be shared.

Identifying the Participants and Their Roles

Wheel company W1 would like to join a bicycle supply chain network. If it joins, it will supply wheel products to bicycle company B1. However, it does not manufacture its own spokes and tyre products. As a result, it will obtain them from companies Sp1 and T1. The wheel company is aware that this supply chain includes other companies that provide various materials, such as rubber, steel, carbon fibre, and frames, for the production of its main product, a bicycle. This implies that this supply chain comprises intermediary products as well. From its perspective, the wheel company identifies the supply chain actors and their roles in the supply chain.

To begin, the wheel company can identify that there is a *trusted third-party* in the supply chain that is responsible for calculating the order lead time of the bicycle supply chain. Second, the bicycle company contacts the wheel company to supply wheel manufacturing; bicycle company B1 is in a *consumer* role with the wheel company. Furthermore, the wheel company does not produce spokes and tyre products and will obtain them from other companies: tyre company T1 and spokes company Sp1. As a result, they are identified as *supplier* companies for the wheel product. There are no competitors since the wheel company is the only one that supplies wheel products to the supply chain. The supply chain comprises *competitor* steel product companies: S1, S2, and S3. The wheel company, however, identifies them as *complementor* companies. The wheel company also identifies the other two participating companies, F1 and CF1, as *complementor* companies since they provide complementary products and services to the supply chain. The following list includes the participants and their respective roles:

- *Trusted third-party*: TP1
- *Consumer*: B1
- *Supplier*: T1 and Sp1
- *Competitor*: None
- *Complementor*: F1, CF1, R1, S1, S2, and S3

Identifying the Collaborations

The wheel company can identify the collaboration as a *network collaboration* since the bicycle supply chain example is a collaboration in which a collection of enterprises works together with common aims and values.

Identifying the Assets

The assets represent the data that needs to be protected. The assets of wheel company W1 can be identified in the bicycle supply chain network as:

- Wheel company W1 shares the data related to wheel production processing time and transportation time. As they are related to business/production processes and trade secrets, *wheel production processing time* and *transportation time* are identified as assets.
- The bicycle supply chain network topology has been made available to all supply chain participants. As a result, wheel company W1 emphasizes *its business relationships* as an asset to be protected in this supply chain.
- The disclosure of the processing and transportation times can impact the *reputation* of the wheel company; it is also considered an asset.

- Lastly, because wheel company W1 may be concerned about disclosing its identity to the whole supply chain, the *membership* information is likewise also considered confidential.

3.6.2 Modelling the Attack and Threat Landscape

From the definitions we provided in our threat modelling approach, wheel company W1 can identify the threat actors and threats to investigate what may go wrong with the collaboration. It can then establish its security and privacy requirements.

Identifying the Threat Actors

Since our threat modelling approach focuses on the internal threat actors, we identify the internal threat actors for wheel company W1. We identify the threat actors based on their relationships with the wheel company. The following are the identified threat actors:

- As a consumer of the wheel company, bicycle company B1 can act as a *downstream threat actor*. Furthermore, the trusted third-party TP1 gets data from companies in order to compute order lead time; it also acts as a *downstream threat actor*.
- The suppliers (T1 and Sp1) and the sub-suppliers (R1, S2, and S3) of wheel company W1 can be identified as *upstream threat actors*. S1 and CF1 companies are also suppliers that can be identified as *upstream threat actors*.
- Lastly, frame company F1 can be identified as a *lateral threat actor* since they share a consumer, B1.

Identifying the Threats and Mapping Them with Actors and Assets

In this section, we identify some types of threats made by the threat actors to the assets of the wheel company. The identified threats for wheel company W1 are listed below, along with how they are mapped to the threat actors and the assets of the wheel company in Table 3.3. It is important to note that likelihood and impact can vary based on factors such as the security measures in place, and the motives of potential attackers:

- *Spooofing:* All participating companies know each other in the bicycle supply chain network. All kinds of internal threat actors may attack the identity of the wheel company W1. An attacker, e.g. F1, sends data to TP1 posing as W1, providing altered lead time estimates for the wheel production. TP1, believing the false information, adjusts the bicycle production schedule accordingly. The impact is high as production schedules are disrupted, resources are misallocated, and company B1 may incur financial losses due to increased production costs and potential delays. While these delays might seemingly aid F1 in achieving timely production and delivery of the frame product, they can equally cause financial losses for F1. Specifically, B1 could lead to a decrease in the rate of orders for sub-products from their suppliers, thus impacting their financial performance. The likelihood of the attack is moderate. Another example is that a supplier, e.g. T1, can want to gain a competitive edge by obtaining information about the other supplier (Sp1)'s product processing time and transportation time. T1, posing as W1, could request access to those specific product details. If Sp1 falls for the impersonation and provides the requested information, T1 can gain valuable insights into Sp1's operations, strategies, and competitive advantages. Both impact and likelihood of the threat are high.

- *Tampering:* To compute the order lead time, the trusted third-party TP1 obtains data from companies in the supply chain (product processing time and transportation time). As a downstream threat actor, TP1 can modify the data of companies throughout the order lead time computation of the supply chain and damage the result. Therefore, the impact of an attack is high. However, because they claim TP1 is a trusted party, the likelihood of this attack is low. Additionally, if S1, as an internal attacker, gains access to communication channels between TP1 and supplier W1, it can send fraudulent messages, altering order quantities and delivery timelines. Incorrect information can lead to disruptions in the supply chain, causing delays in production. Therefore, the impact of the attack could be high. S1, on the other hand, gains no profit from tampering with the data of wheel company W1. Therefore, the likelihood of the attack is low.
- *Information Disclosure:* The trusted third-party TP1, functioning as a downstream threat actor, possesses the capability to expose sensitive details regarding the wheel production processing time and transportation time, both of which are regarded as confidential information. Given the potential consequences in terms of trade secrets and reputation, the severity of this threat is deemed to be high. Nonetheless, due to TP1's established status as a trusted entity, the probability of such an attack occurring remains low. On the other hand, as upstream threat actors, the suppliers of the wheel company might reveal the business processes and relationships they share with the wheel company to one another, enabling collusion. For example, tyre company T1 and spokes company Sp1 might share the business processes data they have with the wheel company with each other. They could make secret bargains, such as determining transportation time and production cost. Both the likelihood and impact of this attack are high.
- *Denial of Service:* Due to the fact that all participating companies use the same network, they might absorb the necessary resources to prevent the wheel company from accessing the services. This attack delays the business processes

of the wheel company. Therefore, the attack has a high impact. Furthermore, an internal attacker can target the underlying infrastructure of the collaboration platform, such as server resources, bandwidth, or memory, causing performance degradation or complete unavailability. This can disrupt lead time management and hinder real-time collaboration. Consequently, the impact of the threat could be high. However, there is no advantage to carrying out these attacks since they also affect the processes in the supply chain, causing delays for internal attackers. Therefore, the likelihood of the attacks is low.

- *Elevation of Privilege:* Frame company F1, as a lateral threat actor, might be interested in gaining access to information about the business processes and business relationships between bicycle company B1 and wheel company W1 in order to analyse its own business processes and business relationships with B1. It can also affect the relationships between B1 and W1. Therefore, both the likelihood and the impact of the attack are high. Another potential threat emerges in the form of TP1, acting as a downstream threat actor. TP1 could potentially exploit a vulnerability to gain unauthorized decision-making authority within the collaborative lead time process. This could result in critical decisions being made without proper evaluation, leading to collaboration failure and causing a high impact.
- *Linkability:* Because the supply chain network structure is accessible to all participating companies, any internal threat actor has the potential to establish connections among the companies and uncover their interrelationships within the network. This form of attack targets the business relationships of participating companies. For instance, consider the case of the steel company S2, acting as an upstream threat actor. In this scenario, S2 could potentially target the connections and data exchanges occurring between Sp1 and W1 to gain insights into the trade secrets held by Sp1, subsequently leveraging this privileged information in its interactions with the same entity. Therefore, the impact of the attack is high. However, this information is just shared with TP1. Therefore, the likelihood of the attack is low. Since trusted third-party TP1 is responsible for calculating the order lead time computation of the supply chain, as a downstream threat actor, TP1 can link not only the business relationships but also the production processing time and transportation time of the wheel company W1. As not all companies get benefits from knowing this data, the likelihood of the attack is medium. However, they might share this data with some outside/inside companies that *are* interested. When business relationships are linked to production processing time, some trade secrets might be revealed. Therefore, the impact of the attack is high.
- *Identifiability:* All participating companies in the bicycle supply chain send their production processing and transportation times to the trusted third-party

TP1. Because TP1 can identify which company provides which data, this violates the anonymity and pseudonymity of companies. The likelihood of the attack is high, but the impact is medium. Wheel company W1 may be willing to take this risk in order to get the desired result. If T1 operates as an upstream threat actor and gains knowledge about the other suppliers of W1, along with the specific data they provide, it could potentially deduce their trade secrets associated with W1. Therefore, the impact of the attack is high. However, this information is only shared with TP1, a trusted third party. Consequently, the likelihood of the threat is low.

- *Non-repudiation:* In cases where a company participating in the supply chain, for instance, W1, provides falsified information about the shared data to others, TP1 can deny such outright information and provide accurate details in the response. The probability of TP1 taking this action is high. This adversely affects the company's reputation and trade secrets, thus resulting in a significant impact. We could encounter another scenario where the suppliers of the wheel company, as upstream threat actors, might claim that they dispatched the materials on time, even if they actually did not do so. In this situation, TP1 can play a crucial role in facilitating a resolution for this disagreement. The impact of the attack could be high if W1 fails to substantiate its non-receipt of the materials, potentially resulting in irreparable damage to W1's reputation. However, considering that the suppliers' supply chain processes would experience disruptions, the likelihood of this threat becoming a reality remains low.
- *Detectability:* The trusted third-party TP1 knows the production processing time and transportation time of the wheel company. TP1, as a downstream threat actor, might detect the production flow of the wheel company based on the number of bicycles ordered by the final consumer FC1. The likelihood of the attack is low since there is no value to knowing the production flow of the wheel company for TP1. On the other hand, the attack can have a high impact on the wheel company, since it may reveal the trade secrets of the wheel company. However, wheel company W1 may be willing to take this risk in order to get the desired result, so the impact could be medium. Another illustrative scenario involves the collaboration between B1, the bicycle company, and W1, the wheel company, which is for producing bicycles. F1 learns about this collaboration through the supply chain network information. F1, as a lateral threat actor, might want to gather information about the collaboration's focus and potential product details. By doing so, they could adapt their own product development strategies accordingly. A successful attack by F1 could potentially expose not only the production processing time and transportation time but also impact the companies' reputation and trade secrets. Consequently, this

threat could cause a significant impact. It is important to note that this sensitive information is exclusively shared with TP1. Therefore, the likelihood of the attack will be low.

- *Content Unawareness:* The wheel company separates its production processing time and transportation time, rather than providing a single lead time for wheel production to TP1. This information can be used by TP1 to benefit other participants. For example, if frame company F1 discovers that W1 is delivering its products later than expected, it may choose to also adjust its delivery schedule and simultaneously supply its products to other supply chains. However, this could affect the timing for FC1 company. Since TP1 would not gain significant profit from an attack, the probability of such an attack occurring is low; however, it would have a significant impact on companies and the supply chain as a whole. Moreover, in the event that computation outcomes are shared without implementing appropriate data protection measures and mechanisms to prevent collusions within the system, a latent vulnerability emerges, ultimately leading to a potential data breach. Such a breach could give rise to serious legal consequences, substantial fines, loss of customer trust, and lasting damage to the company's reputation.

Deriving the Security and Privacy Requirements for the Wheel Company

To join the bicycle supply chain, wheel company W1 requires some security and privacy properties that the supply chain should fulfill. We derive the security and privacy requirements for the bicycle supply chain based on the identified threats. While the likelihood of some threats is low, their potential impact remains high, as they target W1's assets. Therefore, all threats require significant consideration for effective mitigation.

Firstly, the wheel company may agree to share the necessary data (production processing time and transportation time) with TP1 for the order lead time calculation of the bicycle supply chain network. These details, however, should not be revealed to TP1 company as they could disclose W1's confidential information. During the computation, this data should be kept confidential, not only from TP1 but also from other participating companies. Also, sharing the membership and business relationship information could lead to various attacks, including spoofing, linkability, identifiability, and detectability attacks. Therefore, this information should likewise be kept private within the supply chain. While the participating companies may know the supply chain network structure, they should *not* be aware of the companies with which they do not have a direct relationship or the details of collaborations/relationships in which they are not involved.

Table 3.3: Mapping Threats to Threat Actors and Assets

Threat	Threat Actor	Asset	Likelihood	Impact
Spoofing	All kinds of internal threat actors	Production processing time, Transportation time, Membership	M, H	H, H
Tampering	Downstream and upstream threat actors	Production processing time, Transportation time	L, L	H, H
Information disclosure	Downstream and upstream threat actors	Production processing time, Transportation time, Business processes, Business relationships	L, H	H, H
Denial of service	All kinds of internal threat actors	Reputation, Business relationships	L, L	H, H
Elevation of privilege	Lateral and downstream threat actors	Business processes, Business relationships	H, L	H, H
Linkability	All kinds of internal threat actors	Production processing time, Transportation time, Business relationships	L, M	H, H
Identifiability	Downstream and upstream threat actors	Production processing time, Transportation time, Trade secrets	L, H	H, M
Non-repudiation	Downstream and upstream threat actors	Trade secrets, Repudiation	H, L	H, H
Detectability	Downstream and lateral threat actors	Trade secrets, Production processing time, Transportation time, Reputation	L, L	M, H
Content unawareness	Downstream and internal threat actors	Production processing time, Transportation time, Trade secrets	L, L	H, H

L Low, M Medium, H High

Furthermore, some attacks might not be mitigated at the technical level. For example, even though sharing data regarding product processing time and transportation time in plaintext might not be necessary for TP1, it could still require specific knowledge of business relationships to ensure precise computations. Consequently, sharing business relationship information of companies with TP1 becomes imperative. Nonetheless, the necessary mitigation strategies must be implemented at a legal or contractual level.

3.7 Summary

In this chapter, we introduced a threat modelling approach for business collaborations, allowing companies to better understand the risks of joining such a collaboration. In addition, our threat modelling approach can also be used to design and develop secure and privacy-friendly collaborative business networks. We gained a better understanding of the potential risks, attacks, and threats of joining such business collaborations and exchanging data in those collaborations through our threat modelling process. In addition, we learned the requirements, i.e. properties the system should fulfill. In the next chapter, we will look at how the identified threats can be addressed and how a model can meet the requirements. We present a novel scalable, privacy-preserving paradigm for business collaborations that ensures the data confidentiality of the participants in a collaborative business network.

4. Scalable Privacy-Preserving Model for Collaborative Business Networks

After building our threat modelling approach to understand the risks and threats in collaborative business networks and deriving the security and privacy requirements of those networks from our threat modelling, we focus on the security and privacy countermeasures and techniques that address those identified risks and threats and meet the determined requirements. We develop a new scalable privacy-preserving model that makes confidentiality-enhanced computations on shared data in business collaborations. In this chapter, we focus on particular core aspects of our approach, representing how to perform secure computations over shared data in collaborative business networks and addressing the identified security and privacy threats given in the previous chapter. In Chapter 6, we will later present a complicated scenario to cover the complete aspects and contributions of our approach. In more detail, the contributions of this chapter can be summarized as follows:

- We propose a new scalable privacy-preserving model for business networks that ensures secure computations over shared data within business networks, which is based on a combination of decomposing the computation required for collaboration using the hierarchical grouping approach and the use of secure multi-party computation (SMPC).
- We perform a security and privacy analysis of our model that shows how we ensure that our model is secure against the attacks given in Section 3.3 and how we mitigate those attacks.

4.1 Confidentiality-Enhanced Collaborations in Business Networks

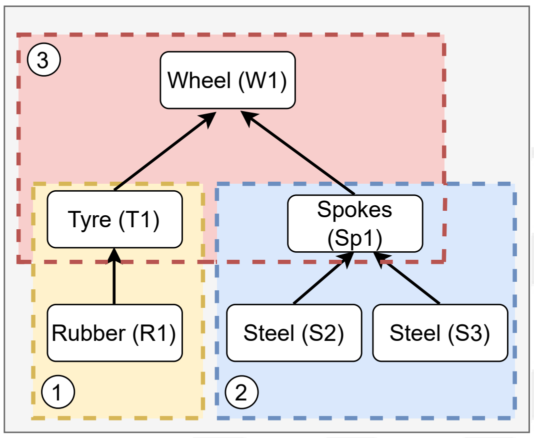
As we discussed in previous chapters, computations in a collaborative business network requiring inputs from multiple parties can reveal confidential information within the network. Consequently, many parties are reluctant to share their data, which is required for close collaborations in general [95, 15, 18, 19]. Although the privacy protection of collaborations and the data being shared and processed in those collaborative business networks are important, there are few existing efficient techniques to preserve the privacy of those networks. In this chapter, we investigate how to integrate SMPC technology to address the security and privacy problems regarding data sharing and processing in business networks containing collaborative executions of business processes. SMPC is capable of being applied to many collaborative scenarios to enable distributed parties to jointly compute an arbitrary function without revealing their inputs and outputs. However, how to implement SMPC in those scenarios to prevent the disclosure of confidential data of parties and collaborations during computations is still an important and challenging problem. This is due to limitations of current technologies, such as performance, storage, and resource constraints.

We present a novel approach that uses SMPC to provide increased protection of the data required to complete a computation in business scenarios that include collaboration among multiple parties. Our aim is to address the security and privacy threats regarding data sharing in collaborative business networks that we identified in Chapter 3. Our approach to improving security and privacy in collaborative business networks is based on three observations: 1. SMPC requires quite a number of messages to be sent between participants; it involves communication and connectivity between all participants. Hence, the performance of computations, using SMPC naïvely, is highly affected by the number of participants within a network. 2. Usually, companies in the network consider their partners with whom they have a direct relationship as confidential. 3. While the number of partners within a network can be large, usually each company only has a relatively small number of partners that they have direct relationships with. These observations led to the idea of *recursive computations* in collaborative business networks. Our approach makes one-level aggregations and local computations in a recursive way that ensures information privacy without causing high computational overhead and high communication costs. In the following sections, we describe the details of our model. We give a partial supply chain example – the supply network of bicycle wheel production, from our running example in Section 2.4 – to explain our model and discuss certain concepts.

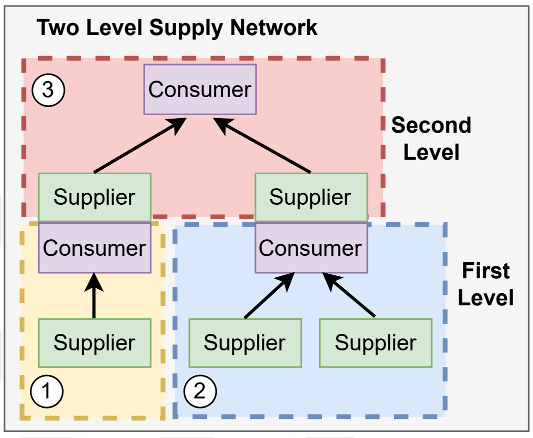
4.1.1 Hierarchical Grouping

We noticed that in collaborative business networks, companies have partners that they directly work with and they also have other partners that they work with indirectly. This means that in a business collaboration, the network has natural groups. We use these natural groups to make the problem simpler, and we call this "hierarchical grouping". In order to enable local computations and aggregations in collaborative business networks, we create hierarchical groups. In our approach, one consequence of hierarchical grouping is to improve the performance of the computations. Although that is not a security or privacy requirement, it makes our approach pragmatically more applicable.

However, our actual aim of making hierarchical groups in collaborative business networks is to minimize the risk of sharing unnecessary data with a large group of companies by working with smaller groups. As we mentioned above, although the number of companies within a collaborative business network can be large, the number of partners with whom those companies have a direct relationship is usually relatively small. We divide companies into groups based on their relationships and their levels in the network.



(a) Hierarchical Grouping in Wheel Prodution Network



(b) Roles of Participants in Groups

Figure 4.1: Hierarchical Grouping of Running Example Supply Chain

Recall the production of the wheel from Section 2.4 (Figure 2.6): six companies form the supply network for wheel production. They provide the following processes: *Steel (S2), Steel (S3), Spokes (Sp1), Rubber (R1), Tyre (T1), and Wheel (W1).* This supply network (collaborative business network) can naturally be divided into a hierarchy based on the "direct supplier relationship" (see Figure 4.1a). We represent how the groups are constituted in Figure 4.1a and Figure 4.1b. Each group of companies has one consumer and several suppliers. For example, *Tyre company T1* and *Spokes company Sp1* in Figure 4.1a supply their products *Tyre* and *Spoke* to *Wheel company W1*. Hence, they make up one of the groups in the network (group 3) as *T1* and *Sp1* companies have a direct relationship with *W1* company. The other groups in this supply network are group 1 and group 2, respectively. In group 1, consumer *Tyre company T1* has just one direct supplier *Rubber company R1*. Group 2 has *Spokes company Sp1* as a consumer and *Steel companies S2* and *S3* as its direct suppliers. Within a large collaborative business network, an individual company can play different roles in different groups in the network. For example, *Spokes company Sp1* is participating as a supplier in group 3 and it has the consumer company role in group 2 (see Figure 4.1b).

4.1.2 Creating a Secure Communication Infrastructure

After making a hierarchical grouping in the network, we establish a secure communication infrastructure. We do this by building a public key infrastructure (PKI). Our aim of building the PKI is to mitigate the risk of disclosing or transferring data to unauthenticated parties. For each group, we establish a PKI using certificates where the company taking responsibility for the group takes the role of the certificate authority (CA). The other companies participating in those groups have a supplier role as they supply information to the group computations. Suppliers in those groups can use pseudonyms when joining a PKI, and, by default, we recommend using different pseudonyms when joining different business networks or if a company participates in different levels or groups of the same business network. The reason we recommend using pseudonyms is to have the pseudonymity property between companies that do not have a direct relationship. In our approach, the main company that takes responsibility for a group also has an authorized certificate besides the CA certificate to join the computation in the group. The important difference between a PKI with hierarchical CAs and our infrastructure is that in our model, the CAs are independent of each other. In other words, there is no common root CA for the complete business network. From the hierarchical grouping of our wheel production supply network example, we derive the communication infrastructure for the supply network as shown in Figure 4.2.

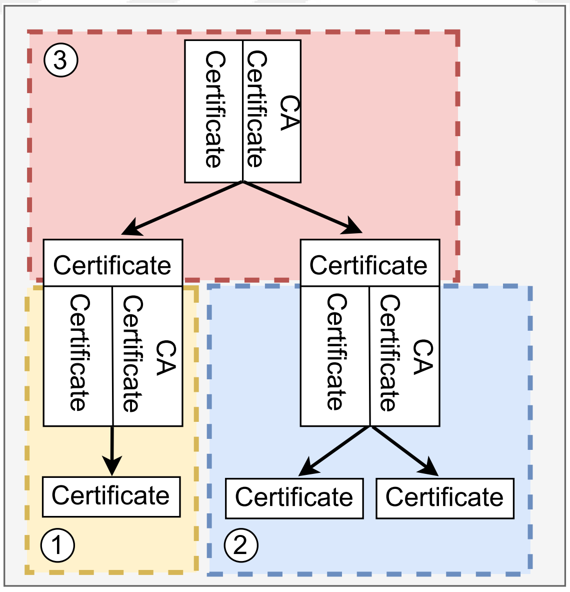


Figure 4.2: Distribution of Certificates in Wheel Production Supply Network

As seen from the figure, the companies that are just in a supplier role, like *Rubber company R1* and *Steel companies S2 and S3* have authorized certificates for the group they joined. The companies (like *Tyre company T1* and *Spokes company Sp1*) that have both consumer and supplier roles in the supply network (see Figure 4.1b) hold several certificates. *Wheel company W1* company has only a consumer role, and it has one CA certificate to authorize the certificates of the suppliers in group 3 and one self-authorized certificate to join the computation in the group.

4.1.3 Integrating SMPC into Hierarchical Groups

In a collaborative business network, we can provide the secure and authentic transfer of data among participating companies through the use of PKI (basically the use of digital certificates) such that outsiders to the network cannot learn anything about the data. However, everyone in the network can still know the transferred data. If participating companies in the network would like to jointly compute a function without revealing their inputs to other members in the network, then they need to have a method to provide that. Secure multi-party computation (SMPC) is a potential solution to this problem. SMPC enables companies to securely compute on distributed data without revealing any individual information. It ensures the confidentiality of data against both outside attacks and inside attacks. Note that although SMPC is the preferred technique for confidentiality of data, it does not provide authentication properties, as does PKI. Both techniques, PKI and SMPC, rely on cryptography, but they have different features and provide different properties. Therefore, in our system, we need both techniques.

After creating hierarchical groups and establishing a secure communication infrastructure for each group, we start to perform secure local computations for each group in the network. The core idea of our approach is to perform local computations for each group in the collaborative business network by using SMPC to protect the confidential information that the companies provide to those groups. The fundamental security property of SMPC is that all participants only know their own input into the joint computation and, if published, the final output. While it is technically feasible to perform SMPC with only two participants, it is generally recommended to involve at least three participants to ensure the privacy and security of the participants' data in the computation. In a two-party scenario, if one participant turns malicious or is compromised, the entire security of the computation could be compromised, as there would be no additional party to help detect and mitigate any malicious activities. This potentially leads to unauthorized access or leakage of sensitive information. With three participants, the protocol can incorporate additional checks and balances, such as threshold mechanisms, which enhance the overall robustness and reliability of the secure multi-party computation process, making it harder for a single participant to collude or compromise the protocol.

As SMPC requires that all participants of the joint communication exchange messages with each other, each member of a group can learn the size of the group and the pseudonyms used for creating the certificates. The detailed information about SMPC is given in Section 2.3. As our collaborative business networks include hierarchical groups, where the result (output) of a group computation is used as an input for another group computation in the network, these computations should be run sequentially and/or in parallel depending on their levels and the function they perform in the network.

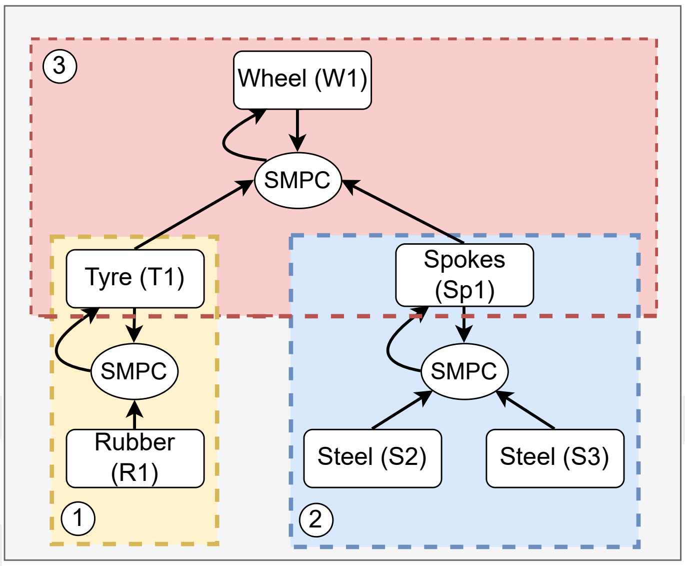


Figure 4.3: Using SMPC for the Computation of the Wheel Production Supply Network

The wheel production supply network (Figure 4.1b) has two levels. The computations for wheel production take place in the upstream direction. We begin by performing local computations on first-level groups. At this level, there are two groups, which are independent of each other: group 1 and group 2. Therefore, they can run their computations in parallel. The computation for group 1 happens between *T1* and *R1*. In this case, we can use SMPC, but ensuring data confidentiality becomes challenging due to the limited number of participants in this group, which consists of only two participants. Specifically, *T1* has a single supplier, namely *R1*. As *T1* receives the computation result, it becomes possible for *T1* to deduce the data provided by *R1* for the local computation. However, we ensure the confidentiality of data by using SMPC for the rest of the supply network. The computation for group 2 is between *Sp1*, *S2*, and *S3*. After *T1* and *Sp1* get the results from their computation groups, they use these results as input for the final computation. The final computation (group 3) happens between *W1*, *T1*, and *Sp1*, and the consumer of this group *W1* gets the result of the full computation for the wheel production supply network.

4.2 Confidentiality-Enhanced Lead Time Calculation of Wheel Production

As is seen from the lead time computation steps of a customer's order in Subsection 2.4.2, there is one big centralized system where all computations over data provided by participants are made in one centre in the supply network. Each company sends its data to the trusted third-party TP1, and the required computations are performed by TP1. One of the main aspects in our approach is to minimize the risk of sharing unnecessary data with a large group of companies by working with smaller groups. Therefore, in the customer order lead time computation example, we first apply this aspect, which is *hierarchical grouping*. We divide the companies into groups where companies have direct relationships and make computations locally within those groups. However, the data that is used in those groups can be sensitive and can include market profits. Therefore, we should also use a technique to secure the data in those group computations. We apply the core idea of our approach, *doing secure local computations for each group using SMPC*, to those groups. In addition, we *establish a PKI* to secure communications within those groups. In the previous section, we represented the security and privacy-enhanced model of the wheel production supply network as an example to explain our model. Now, to show the secure customer order lead computation step by step, we continue using our partial supply network example (wheel production). To show how the process works, we have included a high-level pseudocode of the enhanced lead time computation in Algorithm 1.

The number of groups involved in the supply chain is represented by I, while the number of participants in each group is denoted by n_i. To compute the local lead time for group i, first, each participant enters their respective production processing time (pt) and transportation time (tt) into SMPC. These input values are stored within a local matrix we call LD, as shown in lines 5 to 8 of the code. Subsequently, in line 10, the production processing time of the consumer is allocated to a variable named $consumerProductionTimes$ for future use. Next, the maximum aggregated time, which considers both production processing and transportation times, among the suppliers of group i is determined via SMPC (the lines of the code from 12 to 17). It is important to note that when available and required, suppliers provide their outbound lead time as input. This lead time is a result of the actions taken by the upstream group. Finally, the local lead time for group i is calculated by adding the maximum aggregated time to the consumer's production processing time in line 19. The resulting local lead time for the group is then transmitted to the consumer through SMPC (line 21). It is important to highlight that the consumer saves this result for use in other group computations as outbound lead time.

Algorithm 1 Pseudocode of Confidentiality-Enhanced Lead Time Calculation

```
1: ▷ If there are $I$ groups in a supply chain
2: for $i \leftarrow 1, I$ do
3:     ▷ For local lead time computation of group $i$
4:     ▷ Request their production processing and transportation times as inputs
       from each participant in group $i$, generating local matrix $LD_i$
5:     for $j \leftarrow 1, n_i$ do
6:         Read $LD_i[j, pt]$
7:         Read $LD_i[j, tt]$
8:     end for
9:     ▷ Consumer's production processing time
10:    $consumerProductionTime = LD_i[1][pt]$
11:    ▷ Maximum aggregated time among suppliers
12:    $maxAggTime = 0$
13:    for $k \leftarrow 2, n_i$ do
14:        if $maxAggTime < (LD_i[k][pt] + LD_i[k][tt])$ then
15:            $maxAggTime = LD_i[k][pt] + LD_i[k][tt]$
16:        end if
17:    end for
18:    ▷ To calculate the local lead time of group $i$, the production processing time
       of the consumer is added to the maximum aggregated time
19:    $localLeadTime = maxAggTime + consumerProductionTime$
20:    ▷ The result is output to the consumer of group $i$
21:    Print $localLeadTime$
22: end for
```

In addition, Figure 4.4 illustrates how we apply our approach to the computation of the customer order lead time of wheel production in order to achieve more secure and confidential collaboration. As we determined in the previous section, we start with the computations at the first level, groups 1 and 2. The input for suppliers in a group computation is their production processing time and transportation time (or outbound lead time and transportation time if they are intermediate participants in the supply network), while for the consumer, the input is its production processing time. As group 1 only has two companies, it is impossible to keep *R1*'s data (the total time required for the production processing ($t_{\text{prod}_{\text{rubber}}} = 6$ days) and transportation ($t_{\text{trans}_{\text{rubber}}} = 3$ days) of Rubber) private from *T1*. Similarly, if the result is shared with all computation group participants, then the production processing time of consumer *T1* ($t_{\text{prod}_{\text{tyre}}} = 9$ days) will also be learned by supplier *R1*. The lead time for tyre production is computed as:

$$\begin{aligned} SMPC(R1, T1) &= t_{\text{prod}_{\text{rubber}}} + t_{\text{trans}_{\text{rubber}}} + t_{\text{prod}_{\text{tyre}}} \\ &= 6 + 3 + 9 = 18\,\text{days} \end{aligned} \tag{4.1}$$

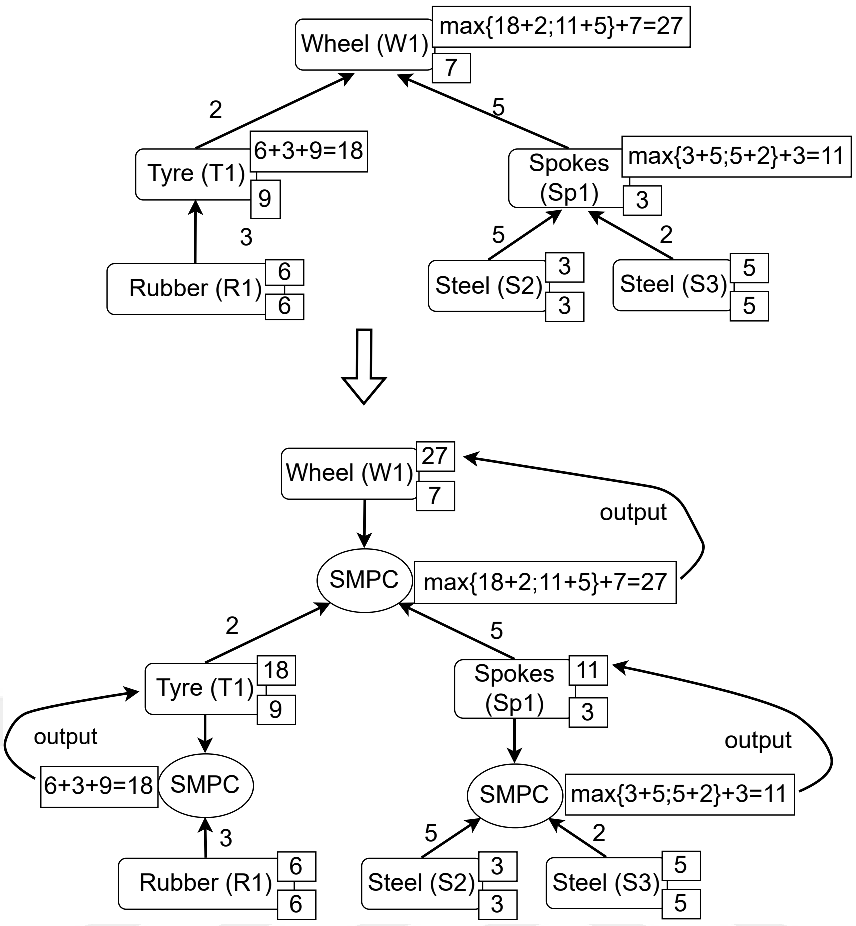


Figure 4.4: Secure Customer Order Lead Time of Wheel Production

In parallel, there is another computation (group 2) happening between suppliers *S2* and *S3* and consumer *Sp1*. They jointly run the function required for the order lead time of spokes by their inputs. Then, consumer *Sp1* gets the result of the local computation, which is the outbound lead time of spokes production (11 days), to use at the upper level in the network. As the computation happens via SMPC and it includes three participants, companies are not able to learn the inputs of other companies in the group. The lead time of the spokes is computed as:

$$\begin{aligned} SMPC(S2, S3, Sp1) &= \max(t_{\text{prod}_{\text{steel(S2)}}} + t_{\text{trans}_{\text{steel(S2)}}}, t_{\text{prod}_{\text{steel(S3)}}} + t_{\text{trans}_{\text{steel(S3)}}}) + t_{\text{prod}_{\text{spokes}}} \\ &= \max(3+5, 5+2) + 3 = 11\,\text{days} \end{aligned} \tag{4.2}$$

After both computations at the first level have been completed and the consumers of those computations get the computation results, which are the outbound lead times of tyre and spokes productions (18 and 11 days, respectively), they join the upper-level computation at the second level (group 3) by using those results as inputs. This second-level computation group includes *T1*, *Sp1*, and *W1* companies. This group runs the final joint computation via SMPC for the wheel supply network, and the result of that computation gives the total customer order lead time for wheel production. The lead time of the wheel production is computed as:

$$\begin{aligned} SMPC(T1, Sp1, W1) &= \max(t_{\text{outbound}_{\text{tyre}}} + t_{\text{trans}_{\text{tyre}}}, t_{\text{outbound}_{\text{spokes}}} + t_{\text{trans}_{\text{spokes}}}) + t_{\text{prod}_{\text{wheel}}} \\ &= \max(18 + 2, 11 + 5) + 7 = 27\,\text{days} \end{aligned} \tag{4.3}$$

4.3 Security and Privacy Analysis of Our Model

We now explain how we prevent or mitigate the attacks and threats given in Section 3.3 in collaborative business networks and how we meet the security and privacy requirements specified in Subsection 3.3.4 by providing detailed security and privacy analysis of our model. In the following, we map security and privacy countermeasures in our model to the identified security and privacy threats in the previous chapter, and we conclude how likely our model is to provide adequate security and privacy requirements. It is essential to emphasize that these countermeasures only partially address the threats, and each one approaches the threats from a different perspective. Table 4.1 represents the threats and the corresponding countermeasures that we use in our model to mitigate them. In the following subsections, we provide a detailed discussion of how the countermeasures used in our model mitigate each type of threat.

Table 4.1: Security and Privacy Threats and Corresponding Countermeasures

Threat	HG	SCI	SMPC
Spoofing	◑	◑	○
Tampering	○	◑	◑
Information disclosure	◑	◑	◑
Denial of service	○	○	○
Elevation of privilege	◑	◑	◑
Linkability and identifiability	◑	◑	◑
Non-repudiation	○	○	◑
Detectability	◑	◑	◑
Content unawareness	○	○	○
Policy and consent non-compliance	○	○	○

HG Hierarchical grouping, SCI Secure communication infrastructure, SMPC Secure multiparty communication; ○ not mitigated at all, ◑ partially mitigated, ● fully prevented.

4.3.1 Spoofing

Spoofing attacks are not our main focus, and our approach does not prevent spoofing attacks. However, we contribute as a side effect: our approach helps to reduce the likelihood of potential attacks. To prevent a company from pretending to be someone else (another company in the network), we need to protect the identity information of companies. In order to do that, in our approach, we first reduce the number of available communication channels between parties by creating *hierarchical groups*. This helps us to minimize the risk of revealing the identity of a company unnecessarily to the large group of companies in the network.

Then, we establish a public key infrastructure for each group where the identities of companies are authenticated by the consumer company that is responsible for the group. And when companies join a PKI, they should use pseudonyms to protect their identity and anonymity. The consumer company that operates the PKI still might know the participants. However, it is a requirement and norm action that the identity of companies in that group be known and authenticated by the consumer company, as they will eventually have an official or legal contract or relationship together. The consumer company distributes the authenticated certificates of companies in the group to all companies in the group as they run a joint computation together, but the identities of the companies are kept secret from each other with the pseudonymity feature.

One can argue that using pseudonyms actively encourages spoofing attacks. However, in our approach, we establish two types of PKI infrastructure to prevent this problem. The detailed explanation will be given in Section 5.2.

4.3.2 Tampering

Our approach prevents tampering attacks against the data in the jointly executed programs. To mitigate the modification of data or code in the jointly executed programs of collaborative business networks, we use SMPC and authenticated certificates (PKI). In our model, the required computation code of a group is compiled by the consumer company of that group, and this version is shared with the other companies that joined the group to run it together in a distributed way. If a participating company, as an attacker, changes something in that compiled code, it will need to compile the code again and share that version with the other companies to be able to run that modified code. As we have a secure communication infrastructure, PKI, in our model, companies will know that this is coming from another company rather than the consumer. Therefore, they will not accept running that version, and the attacker will fail to run that modified computation code.

Also, companies share their data with other companies in the group to run the joint computation together via SMPC. A company, as an attacker, can attempt to modify the data of the other companies, but this data is already in a secret form. Therefore, the attacker cannot modify this data. However, if an attacker would like to destroy or manipulate the computation, they can do so by modifying their own input data provided to the computation. Although the consumer company may find out from analysing the output of the computation, we do not have any countermeasures to check the integrity of the input data. Our model could be combined with a commitment scheme that address the issue of participants supplying the wrong inputs. Commitment schemes allow parties to commit to a chosen data set (or chosen statement) while keeping this data hidden from other companies in the group, with the ability to reveal the committed value later. One of the important applications of commitments that is suitable to our approach is to be in a verifiable secret sharing group, which is a critical building block of secure multi-party computation. A secret is distributed along with commitments to the individual shares in a verifiable secret sharing method. The shares enable each party to verify whether their shares are correct, while the commitments reveal nothing that can help a dishonest group [132].

4.3.3 Information Disclosure

All three countermeasures that we have in our model help to mitigate attacks related to the disclosure of information in collaborative business networks in different ways. First, the hierarchical grouping setup ensures that consumers, which are the companies that consume the data coming from other companies, can only communicate with their direct suppliers, which are the companies that provide the data, and prevents the consumers from learning anything about indirect suppliers (downstream). Similarly, a supplier cannot learn anything about its customer's relationships (e.g. whether the consumer also supplies anything to another consumer (upstream)).

Allowing companies to use pseudonymous handles for their certificates minimizes the risk that companies that have indirect relationships with each other in the same group learn about each other. A company could further obfuscate their participation (this requires cooperation by the company acting as CA), e.g., by taking on the roles of multiple companies. By adding artificial companies using pseudonyms, the consumer company (having responsibility for the group) can, moreover, minimize the risk of revealing the number of participants in the group.

To mitigate attacks on the confidentiality of data in groups or the internal data details of companies (the individual data that they provide to the computations), we mainly rely on the application of SMPC. In general, this gives us a guarantee that companies can only learn their own input into the computation they are involved in and potentially the final output.

However, SMPC does not make any statements about what can be inferred from obtained information. Therefore, it can still reveal some confidential information that should be kept secret. Clearly, if only two companies are involved in the computation or, similarly, only a few data sets are considered, knowing one's own inputs and the final result might allow one to compute the input of the other company. To minimize this risk, as a system requirement, it is important to check that each group executing a computation has at least three participants. Similar checks could be added to ensure that a minimum amount of data is provided by companies. However, there is no one solution that works for all cases; we must examine each one individually. Different collaborative business networks can have different participation types and different computations. We need to think about the formula being computed to understand the minimum number of participants that guarantees that the inputs cannot be inferred.

4.3.4 Denial of Service

In our approach, we do not have a central system that is a single point of failure. We have a distributed system that runs the SMPC application that allows companies in a group to run a joint computation (in a distributed way) where all participants must attend. However, the SMPC protocol also has deadlocks if a company goes down. And this could have a worse effect than the central system.

In a centralized system, if one of the companies stops functioning, it may still be possible to recover. The central system can remove the faulty company, if it is feasible to do so, and continue the computation with the remaining companies. However, in our approach, if a company refuses to participate or provides degraded input, it will affect all the other companies in the group, leading to a failure in the computation. When the computation fails, the company that is responsible for the computation group can recognize and restart the process, but it might not know which party caused it. To prevent or mitigate these types of attacks, an intrusion detection system can be used [133]. Intrusion detection systems are designed to detect traffic irregularities associated with an attack's execution [134]. An intrusion detection system can help to detect which party caused the failure of the computation. In our approach, these types of threats are out of scope and we do not provide any countermeasures to prevent or solve them.

4.3.5 Elevation of Privilege

In our model, the authorizations of companies in a collaborative business network are identified by the roles that they have in the network. We specify the roles of parties in the network by creating hierarchical groups. In each hierarchical group, we establish a PKI for the computation infrastructure. This PKI identifies the roles of participants in the group and provides signed certificates for secure computations. According to the determined authorizations, each company can only join and run the computations of groups in which they are involved; this is ensured by having an independent PKI for each group.

A supplier could claim to be a consumer company of a hierarchical group. We prevent this attack by building a second PKI infrastructure, which is the PKI for the communication infrastructure. This PKI infrastructure allows suppliers to communicate with the intended consumer company and build their computation infrastructures. The details are provided in Section 5.2.

During the computation, companies cannot change, modify, or reveal any data that does not belong to them. This is ensured by SMPC. The threshold aspect of SMPC ensures that no single company has access to the complete input data, and that the computation can only be performed if a specified threshold of companies agree to participate. Consequently, an attacker cannot modify the data unilaterally but must cooperate with others to achieve their goal. If the attacker can increase the number of fraudulent companies beyond the threshold, they may gain unauthorized access to or modify data. To minimize this risk, we recommend setting the threshold for computation as high as possible.

4.3.6 Linkability and Identifiability

As identifiability is a special case of linkability threats, we analyse those threats together. To mitigate linkability and identifiability threats in a collaborative business network, we use all three countermeasure steps in our model. Making hierarchical groups manages the communications and computations between companies in the network. As hierarchical grouping minimizes the number of communications that companies have and reduces the number of companies that can participate in computations, it helps to reduce the links between companies and the risk of revealing their identity to a minimum. In order to mitigate the linkability and identifiability attacks related to the data of companies, we use SMPC as a countermeasure. Since the data is encrypted in SMPC, this reduces the possibility of identifying individual data and linking it with companies.

In collaborative business networks, there can be some groups (hierarchical groups) that include competitive companies. If a company knows that there is just one other competitive company that provides the same data type to the group computation, when the result of the computation is released, it can subtract its data and then link the rest of the result with the other competitive company. In order to mitigate this type of attack, we recommend that companies use pseudonymous certificates when joining a PKI of a group. Also, we highly recommend using different pseudonyms for each group or network that they join. This prevents an attacker from identifying companies and linking them up with their data and the groups that they participated in.

4.3.7 Non-repudiation

This type of threat provides attackers with information about what a user knows, has done, or has said. In our approach, the consumer company that takes over the role of the certificate authority (CA) within a group knows which participants were involved in the group and the computation. Therefore, if a company within the group denies its involvement in a computation, the consumer company can refute such claims. However, since we offer suppliers the option to use pseudonymous certificates, a supplier can deny its participation in a computation to another supplier company within the group. This implies that a company can have the plausible deniability property to companies that are not its consumers.

Also, companies protect the privacy of their data by providing it to computations in a secret form (SMPC). With SMPC, no company can claim or access the data provided by other companies in the computation, thus ensuring plausible deniability of the data inputs. However, this also poses a security threat to the system, as secret form can allow companies to cheat by providing incorrect data and denying responsibility. To address this issue, commitment schemes can be adopted, allowing companies to commit to chosen data or statements while keeping them hidden from other companies in the group. This enables the companies to later reveal the committed value, mitigating security threats. However, such schemes are beyond the scope of our approach.

4.3.8 Detectability

To prevent the detection of any existing IOIs (items of interest, such as subjects, messages, actions, etc.) in collaborative business networks, we take multiple steps. First, to reduce the possible number of IOIs available to an attacker, we divide the companies into hierarchical groups. However, these reduced numbers of IOIs are still visible.

Therefore, as a second step, we use local PKIs for each group separately, and companies use pseudonyms to make their identities undetectable by other participating companies except the consumer. They can be detected by the consumer company, with which they have a direct relationship. This is inevitable because they will most likely have an official contract together. One can argue that companies also have a direct communication with each other during the computation. However, they are using pseudonyms during the computation. Therefore, they are not detectable. Since they are not detectable, their actions are also undetectable. To prevent the detection of the data during computations, we use SMPC. In an SMPC setting, the inputs (data) of companies are kept as a secret, preventing attackers from detecting any messages or data in the computation.

4.3.9 Content Unawareness

This type of threat is related to information disclosure threats. In our approach, the formula being computed, the parameters being given, and the companies participating are determined by the companies that have the responsibility for the groups of collaborative business networks. However, the other companies who joined these groups are aware of certain things, such as what formula is being computed, how many participants are in the computation they joined, which types of data are provided by participants, how this data is processed, etc.

On the other hand, they are not aware of who the other participants are, what specific data values they provided, and so on. This helps parties to be confident in certain aspects, and they are provided by SMPC. Nevertheless, SMPC cannot make any statements about what parties can infer from their input data and the output of a computation. These threats are most likely specific to cases or scenarios, and our approach does not provide a solution for them.

4.3.10 Policy and Consent Non-compliance

Policy and consent non-compliance refers to situations where organizations violate established policies or fail to obtain proper consent for certain actions. These attacks typically involve non-technical aspects, such as intentional or unintentional disregard for rules, regulations, or consent requirements. While SMPC can provide privacy and security benefits, providing a secure framework for data computations, it may not directly address policy and consent non-compliance attacks. Business collaborations need to adopt a holistic approach that combines legal, organizational, and technical measures to address these challenges effectively. Some of these measures may include monitoring policies [135, 136], using policy communication [130] and policy enforcement tools [131, 137].

4.4 Security and Privacy Analysis of the Running Example

We now explain how we mitigate the attacks (Subsection 3.6.2) identified in our running example. Mitigations of attacks that reveal the membership and business relationships of wheel company W1 within a supply chain are mostly built on two pillars: 1. Our approach eliminates the need for a trusted third party. The hierarchical grouping setup ensures that consumers can only talk to their direct suppliers, preventing them from learning (down-stream) anything about indirect suppliers. Similarly, a supplier cannot learn anything about the customers of their customers (up-stream). As a result, only companies that have a direct relationship with the wheel company are aware of its existence, but they are not aware of the wheel company's relationships with other companies in the supply chain. 2. Allowing companies to use pseudonymous handles for their certificates minimizes the risk that suppliers to the same consumer learn about each other. This means that the frame company F1 is not aware of the existence of the wheel company or its relationship with the bicycle company B1.

To mitigate attacks on the confidentiality of data or internal production details of the wheel company, including production processing and transportation times, we mainly rely on the application of SMPC. In general, this gives us a guarantee that a company only knows their own input into the computation and potentially the final output.

4.5 Summary

In this chapter, we have built a model for business collaborations that minimizes the amount of confidential data that needs to be shared between the participants of the business collaboration (improving the confidentiality of the data process within such systems) while, at the same time, offering scalability in terms of performance. We believe that this combination can enable closer collaborations within business networks in general. In particular, this combination can also enable getting the precise and real-time results that are necessary for business collaborations that are time-dependent while maintaining information privacy. On the other hand, as we stated previously, SMPC ignores the knowledge that participants in a business collaboration can infer from information gained from one or more computation results. We will leave these concerns for future work. In the next chapter, we will go into more detail on the approach and provide the implementation details of our model.

5. Implementation of the Scalable Privacy-Preserving Model

In the previous chapter, we first built a scalable privacy-preserving model that provides secure collaborations over shared (confidential) data in business networks. Then, we carried out a security and privacy analysis of our model. This chapter has very close links to the previous chapter. However, it gets into the implementation details of our model.

The model we designed (Section 4.1) for collaborative business networks has a hierarchical grouping approach. The hierarchical grouping approach contains recursive computation groups where the computation results of downstream groups are used as inputs into the computations of upstream groups of a business network. Each group of companies has one consumer company and several direct suppliers. For our implementation, we assume that those hierarchical groups are determined by the consumer companies of those groups, as usually there is a contractual relationship between consumers and their direct suppliers. Also, we put more focus on the implementation of secure communications and collaborations/computations within the hierarchical groups of a business network. In this chapter, we will first give detailed information about the components required to implement our model. Then, we will link the technical decisions that we made to the security requirements we derived in Subsection 3.3.4. Finally, we will present the complete implementation steps required to achieve security- and privacy-enhanced computations in hierarchical groups of a business network.

5.1 Setting up System for Hierarchical Groups

In our implementation, before running any computation in a hierarchical group, we need to set up the system. During system setup, we have a *client-server* architecture as the network communication model. The server is the consumer of the group, whereas the clients are the suppliers of the group, which are the direct suppliers of the consumer. The consumer hosts, delivers, and manages most of the resources and services requested by the suppliers, such as signing certificates, distributing the certificates and the data needed for executing computation programs, and so on. In Section 5.4, we will give a detailed explanation of the resources and services that the consumer company provides.

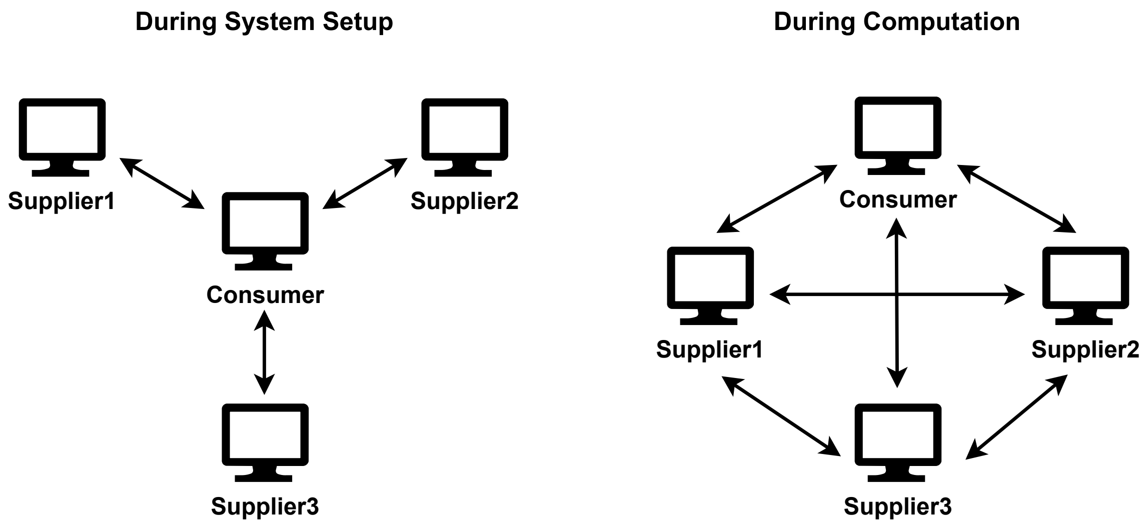


Figure 5.1: Network Communication Architectures in a Hierarchical Group

The consumer and the suppliers exchange messages in a *request–response messaging pattern*. On the server (consumer), we expose a set of services that are accessible via the *HTTPS protocol*. The client (supplier) then directly calls the services by sending HTTPS requests. To formalize the data exchange, we implement a *RESTful API*. In order to provide communications security over the network (provide authentication between the consumer and the suppliers), we set up a PKI. With the PKI, we do everything over mutually authenticated Transport Layer Security (TLS). TLS is a cryptographic protocol that ensures secure communication over the internet by encrypting data transmitted between servers and clients. Then, we integrate SMPC into the group to secure the data being transferred and processed during computations within the group. It is important to highlight that we have a centralized network during setting up the systems of the hierarchical groups. However, when an actual computation happens, we use a decentralized system (*peer-to-peer network*) in those groups. In group computations, there is no specific client or server: all companies, including the consumer company, send and receive data directly to/from each other. Figure 5.1 shows the network communication models that companies have during system setup and group computation.

5.2 Creating the Public Key Infrastructure (PKI)

In order to create secure communication channels between companies in a group, we establish a *PKI*. To ensure a secure configuration of the PKI, it is important to use a public key cryptography scheme that is widely recognized as secure and recommended by experts in the field. At the time of writing, NIST [138], one of the recognized standardization bodies in this domain, suggests the adoption of *elliptic-curve cryptography (ECC)* based on curves specified in NIST Special Publication 800-186 [139], such as the curves P-256, P-384, P-521, among others, which are widely recognized as secure choices for cryptographic operations. These curves provide a good balance between security and efficiency for most practical applications. However, we recommend consulting the latest guidelines and recommendations from NIST or other relevant organizations to ensure the selection of an appropriate secure public key cryptography scheme for your specific use case. To define the format of public key certificates, we use *X.509*, which is an International Telecommunication Union (ITU) standard. An X.509 certificate uses a digital signature to bind an identity to a public key. It contains information about the identity to which the certificate is issued (in our case, this is *Common Name*) and the identity of the issuer. It also includes other standard information, like serial numbers, algorithm information, etc., but these are not our focus. Our main focus with an X.509 certificate is the identity security of the company that owns the certificate. In the following, we will explain how we ensure the identity security of companies in certificates.

In our implementation, we establish two PKI infrastructures. The first is for the communication infrastructure. The aim of the communication channel is to create secure communications between a consumer and its suppliers in the group. This channel is used for *system setup*. During the system setup, the companies, including the consumer in the group, use digital certificates signed by *a public and well-known certificate authority*, such as Google Certificate Authority. The digital certificates in this infrastructure do not have to be signed by the same public certificate authority. However, companies may have to use their real name because of the policy of the public certificate authority. Figure 5.2 presents the PKI of one group for the communication infrastructure during system setup.

Using a well-known CA for signing the certificates of participating companies in the group is important to mitigate potential spoofing attacks on the communication network. For example, if the consumer company uses a self-signed certificate to provide HTTPS services, suppliers who communicate with the consumer cannot be certain whether they are connected to their intended destination. A malicious third party that knows the holder name of the consumer company could redirect the connection using another self-signed certificate bearing the same holder name. Even though the connection is still encrypted, the intended destination is not reached. Therefore, it is important to use a public well-known authority to issue the certificates of the groups during system setup.

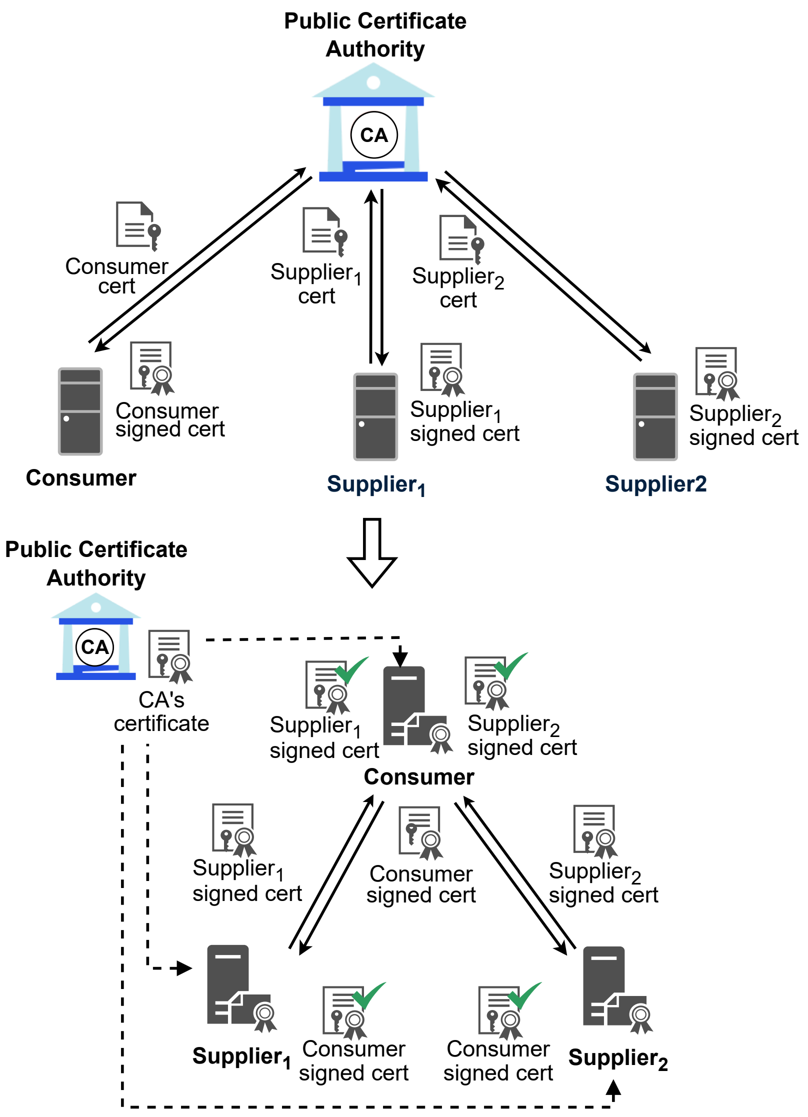


Figure 5.2: PKI for Communication Infrastructure

The second PKI is established for the computation infrastructure of a group and primarily relies on contractual relationships between participants in the collaboration. The aim of the computation channel is to create a secure local computation between participating companies of the computation. This is peer-to-peer network. In this infrastructure, the consumer company of the group takes over the role of the CA for the group. Suppliers trust their consumer company as a CA because they already have a contractual relationship with it. A contractual relationship offers additional assurances, such as identity verification and legal accountability. Therefore, the trust dependency limitation of PKI is less problematic in our approach. The consumer company first creates a local CA key pair that it will use for signing the certificates used during the computation. The consumer company can use its real name/identity in the CA certificate because the identity of the consumer company is already known by the suppliers in the group; it has a direct relationship with each of them. After creating the local CA, the consumer company shares this local CA with its suppliers in the communication network.

Next, companies (suppliers) that would like to join the group computation create a key-pair to use in the computation network. Then, they send a signing request to the certificate authority (the consumer) of the group for authentication. The consumer company issues the digital certificates of the suppliers and stores them for computations in the group. When executing the CA locally within the group, if the CA is not available, then the entire computation in this group gets blocked. However, this aspect of reliability is at least not worse than it was before. While we do not have a solution for this problem, it is less critical since it is limited to the subgroup.

To ensure that their identities remain anonymous within the group, supplier companies would use *pseudonyms* for their common names (CNs) when joining the PKI of the computation group. They agree on these pseudonymous during system setup; the consumer company has direct relationships with all suppliers in the group during the system setup, and the supplier companies trust the consumer company to keep their identities secret (they have a contractual relationship). When a supplier sends a certificate signing request with a pseudonym, the consumer, as the local CA, verifies and signs it and then sends it back to the owner. This process can help to address the anonymity weaknesses of PKI during computation. These steps are made via mutually authenticated channels created during system setup. Figure 5.3 presents the PKI of one group for the computation infrastructure. However, it is important to note that if suppliers want to use pseudonyms, using a different internet service provider (ISP)/endpoint is effectively required, as otherwise supplier's IP will reveal them. A detailed explanation will be given in the next section (Section 5.3).

By default, we recommend using different *pseudonyms* when joining different business networks or if a company participates in different groups of the same network. This ensures that only the consumer company knows its supplier companies, but a supplier company cannot learn the real identity of the other supplier companies within the same group or network. Also, SMPC implementation requires that when supplier companies create their key pairs, they should have *different CNs* (this is supported in our approach), and the CNs should contain *no spaces.* Therefore, we follow this requirement in our implementation.

The publicly well-known authority used in the PKI of the communication infrastructure does *not* mean a common root CA for the complete business network. We use it for the authentication of participating companies of the groups, but it does not sign any certificates used for computations in those groups. The suppliers' certificates in a group are signed by the local CA of that group. There is no PKI with hierarchical CAs; the CAs of each group are independent of each other. They are shared with participants in the secure communication networks during system setup. Therefore, suppliers in a group trust the local CA of the group and also the certificates that are issued by the group CA.

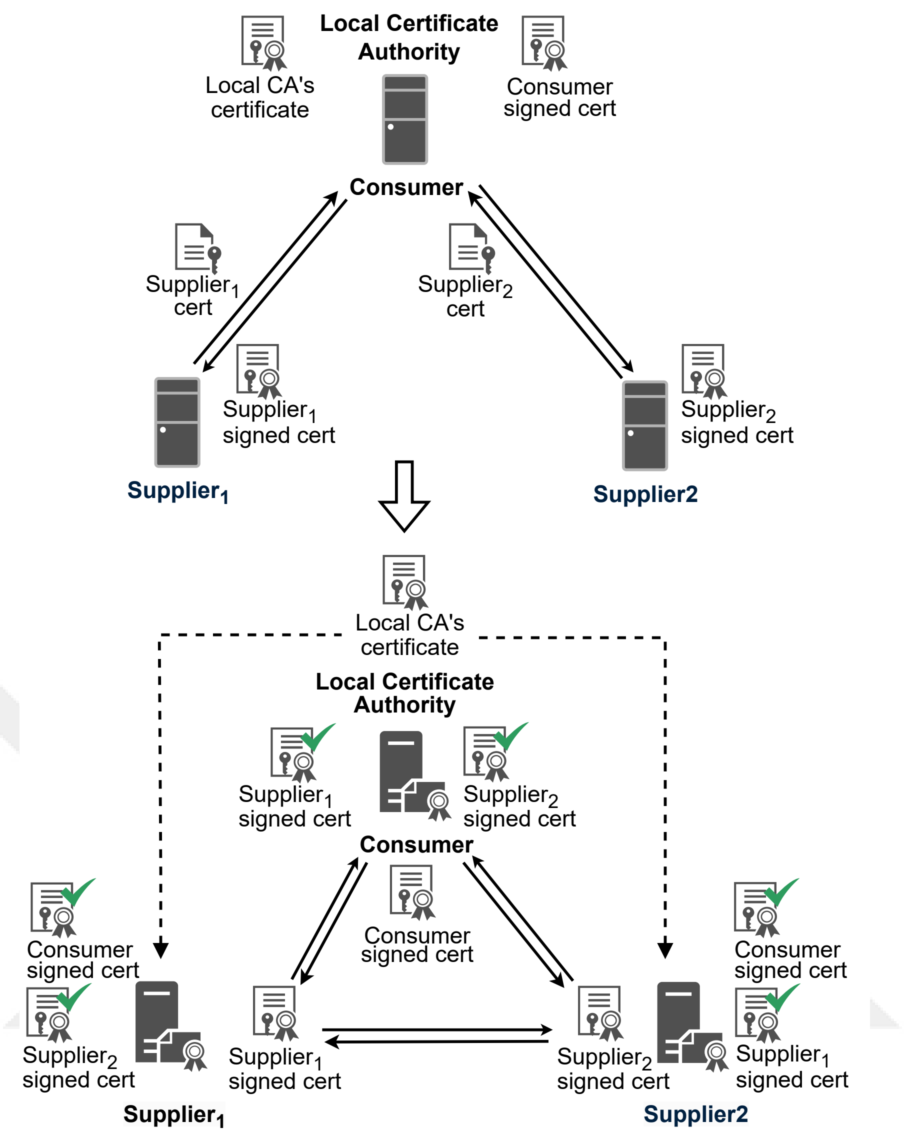


Figure 5.3: PKI for Computation Infrastructure

Not only suppliers but also consumer companies supply data for the computations in a group. Therefore, to participate in a computation, the consumer company also creates a key pair with the same requirements as the suppliers. Figure 5.4 shows the establishment of the group PKI for the computation network of a hierarchical group.

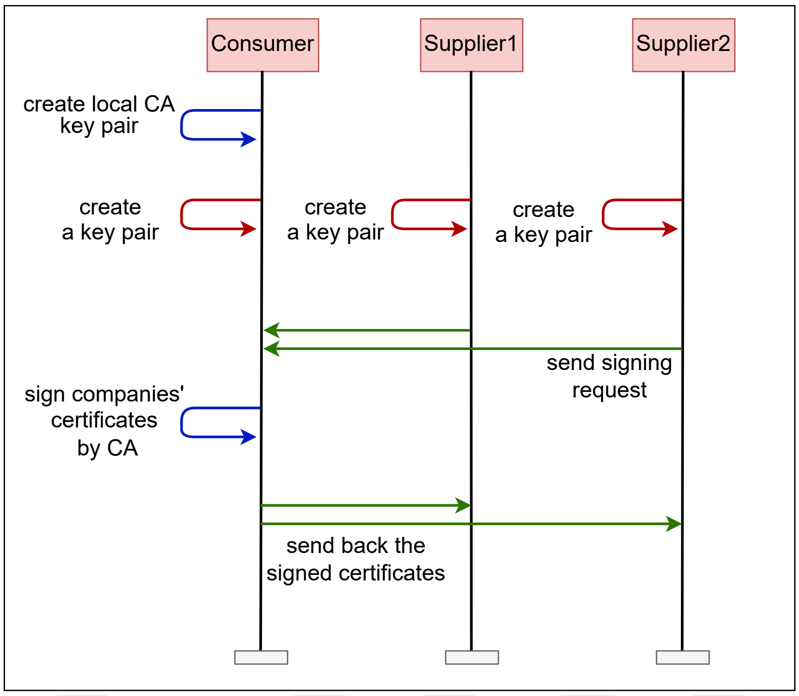


Figure 5.4: The Establishment of the Group PKI

5.3 Integrating Secure Multi-Party Computation (SMPC) into the Groups

To ensure the security and confidentiality of calculations performed on shared data within a group, we use *SCALE-MAMBA* [140] as our SMPC implementation. SCALE-MAMBA is a framework for multi-party computation that allows for secure secret sharing and secure computations among parties. It is primarily written in the *C/C++ programming language* and provides a *Rust-based programming language* for writing distributed computation programs. More information about the working principles and installation of SCALE-MAMBA can be found in [141].

Before performing any computation programs within a group, SCALE-MAMBA requires setting up the system for the computations. This is a one-time setup that includes configuring the networking and/or secret-sharing system being used, as well as setting up a garbled circuit (GC) to linear secret sharing scheme (LSSS) conversion circuit if necessary. In our implementation, as the collaborative business network is organized into hierarchical groups that perform local group computations, we execute the SMPC setup step of SCALE-MAMBA separately for each hierarchical group with the specified parameters.

The SMPC setup of a hierarchical group is performed by the consumer company of the group, as it is trusted by the suppliers and is responsible for the group. If there are changes to the suppliers in the group (such as companies leaving or joining), the setup for the group must be run again to configure the new networking and secret-sharing system and/or set up the GC to LSSS conversion circuit. When the SMPC setup is complete, various types of data are generated, including data for networking, data for secret sharing, and/or conversion circuits, which must be shared with all participants in the group to enable the execution of the actual computation. In the following sections, we will discuss the important parameters that need to be configured during the setup, the information that is shared between participants, and how these factors impact the security of the participants.

Data for Networking

SMPC is a joint computation where suppliers not only communicate with the consumer but also with the other suppliers in the group. They need to know the networking information of both the consumer and the other suppliers in the group. Therefore, first of all, the input provided by the consumer company generates data which includes networking information for the group:

- *the root certificate name* (CA certificate of the consumer);
- *the number of companies* that have joined the computation group;
- and for each company
  - the *IP address* that is going to be used;
  - the *name of the certificate* for that company.

As we mentioned in the previous section, suppliers use pseudonyms for common names in their X.509 certificates in our approach. Therefore, the identities of suppliers are not disclosed via their certificates/certificate names. However, to provide communication between suppliers, they also need to share their IP addresses with each other. IP addresses also have the risk of disclosing identity and tracking companies on the internet.

When a company connects to the internet through an internet service provider (ISP), the ISP assigns a unique IP address to the company's device. This IP address is used to identify and route data between the company's device and other devices or services on the internet. If a company relies on just one external ISP for all its business activities, its IP address might reveal its identity. To mitigate the risk of IP address-related attacks targeting a company's identity during business collaborations, we recommend using different ISPs for these collaborations, separate from those used for their regular business operations. By using different ISPs, it becomes more difficult for attackers to trace the origin of the communication back to the company. Alternatively, implementing virtual private networks (VPNs) can also enhance online privacy. A VPN secures the company's IP address by masking it with the IP address of the VPN server to which the company is connected.

Data for Secret Sharing

In order to securely compute a computation in a group, we use secret sharing schemes, which are methods for distributing a secret among a group of companies in such a way that each of them contributes a share of the secret. The secret can only be reconstructed when a sufficient number of parties' shares are combined together; otherwise, parties cannot derive any intelligible information about the secret.

To set up the secret-sharing system being used, the participant companies in a group agree on the secret-sharing method, and then the consumer company defines the secret-sharing method and specifies the configuration and the following parameters of the chosen method. All companies participating in the group computation have access to the details of the secret-sharing system being used. SCALE-MAMBA offers four secret-sharing methods: *Full Threshold* (as in traditional SPDZ) [142], *Shamir Secret Sharing* (with t < n/2) [143, 144, 145], *Q2-Replicated Secret Sharing* [144, 145, 146], and *Q2-MSP Programs* [146, 147]. Choosing which secret-sharing scheme to use depends on the requirements of the group. Each of these secret-sharing schemes provides a different level or type of security to the system.

- In a *Full Threshold Scheme*, n parties carry shares s_i of a secret s, and all shares are required to reconstruct the secret. This protocol is secure against an adversary that is passive and information-theoretic. Although this scheme provides strong security, it can become impractical as the number of shares increases. Also, SCALE-MAMBA requires a modulus that is compatible with the FHE system they are using. As they use the FHE system for the full threshold scheme, this can cause suboptimal performance.
- *Shamir Secret Sharing* is a form of Linear Secret Sharing Scheme (LSSS) that gives parties the free ability to locally compute linear functions of their secrets. Shamir secret sharing also divides a secret s into n shares (s_1,...,s_n) (n is the number of parties) such that any t, which is the threshold, or more of the s_i shares are sufficient to reconstruct the secret s. However, $t - 1$ or fewer shares provide no information about secret s [143]. This scheme provides information-theoretic security, and therefore, it satisfies the perfect secrecy property. SCALE-MAMBA uses an online phase for Shamir setting that uses the reduced communication protocols of [145]. These protocols do not use a complete communication network for the most costly part of the computation. The reduced number of communications helps to improve the performance of the computation.
- *Replicated Secret Sharing* is another popular LSSS scheme. It was introduced to perform multiplications of secret values. In replicated secret sharing, each party gets not one but several shares of the secrets, and without having all the shares, parties have no information about the actual secrets [148, 149]. In SCALE-MAMBA, to use Replicated Secret Sharing, it is required to enter a complete monotone *Q2 access structure*. The reason for having a Q2 access structure is because when it comes to threshold structures, Replicated Secret Sharing is usually less efficient than Shamir Secret Sharing [140]; it does not scale well with the number of parties [150]. However, any access structure

can be represented with Replicated Secret Sharing. An access structure is a simple way to inform the protocol of which parties can retrieve the secret after the computations are done. In an access structure, there are groups (sets) of parties that definitely need to be able to retrieve the secret. This is called being *qualified.* If a set shoud not retrieve the secret, it is called *unqualified.* One of the things required from an access structure is that it is *monotonous.* This means that no subset of the qualified sets can retrieve the secret. However, any superset has to be able to retrieve the secret. Nevertheless, there can be a set for which that decision comes naturally. If there is no union of two qualified sets that constitutes the whole group, this access structure is called a *Q2 access structure* [146].

- *Q2-MSP Programs* are another way of entering a Q2 access structure, but the structure is entered via a general Monotone Span Programme (MSP) or, alternatively, through the matrix that establishes an underlying Q2 LSSS. Q2 MSP is not itself multiplicative. Therefore, the SCALE-MAMBA tool automatically extends this scheme into an equivalent multiplicative MSP. A Monotone Span Programme is a method of cryptography that involves dividing a secret into multiple shares, which are then given to different parties or individuals. This technique is used to protect the secret and ensure that it remains secure. In contrast to Shamir's Secret Sharing, Monotone Span Programmes allow for the use of any monotone access structure. A monotone access structure refers to a system of rules that determines which individuals or parties are allowed to access the secret, based on the shares they possess.

In our approach, hierarchical groups do not include any specific qualified group of companies that are granted access to retrieve the secret; everybody has the same right to access the secret. As we do not have an access structure in our approach, we eliminate the choices that use an access structure: *Replicated Secret Sharing* and *Q2-MSP Programs.* We look for secret sharing schemes that can be implemented in threshold systems. Also, one of the aims of our approach is to enhance the performance of the system by having performance-efficient computations. As we stated above, the *Full Threshold Scheme* provides strong security, but it can become impractical as the number of shares increases. These required properties limit the choices in secret sharing schemes that we can select from SCALE-MAMBA. The secret sharing scheme provided by SCALE-MAMBA that best fits our approach is the *Shamir Secret Sharing Scheme.* Therefore, we used this scheme in our implementation, and we followed the requirements of SCALE-MAMBA to run this scheme.

SCALE-MAMBA allows us to select the threshold for our SMPC. However, they state that this threshold must be between 0 and $n/2$, where n is the number of participating companies. According to the discussion of SMPC in Subsection 2.3.2, in order to ensure fairness and guarantee the delivery of the output, we must have a guaranteed honest majority, meaning that the number of corrupted parties must be less than half of the total number of participating companies. As long as these conditions are met, SMPCs can be implemented for any function with both computational and information-theoretic security, provided that the parties also have access to a broadcast channel. Therefore, we chose the threshold to be $n/2 - 1$.

5.4 Implementation Steps of Security- and Privacy-Enhanced Computation in a Hierarchical Group

In our implementation, there are four fundamental steps that need to be followed to achieve security- and privacy-enhanced computation in a hierarchical group, as shown in Figure 5.5. First, two public-key infrastructures (PKIs) are set up (1) to be used during the system setup and the computation. The aim of these PKIs is to create secure communication channels between participating companies of the group in both system setup and computations. The details are provided in previous sections. The system setup PKI is used for setting up the SMPC and distributing the data necessary for the computation. The second PKI is used for executing the SMPC program securely.

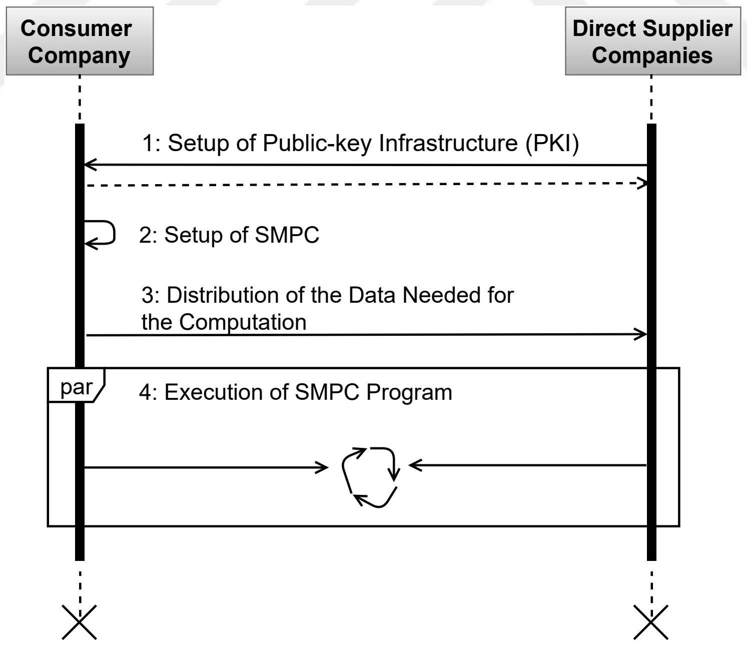

Consumer Company
Direct Supplier Companies
1: Setup of Public-key Infrastructure (PKI)
2: Setup of SMPC
3: Distribution of the Data Needed for the Computation
par
4: Execution of SMPC Program

Figure 5.5: Security- and Privacy-Enhanced Computation of One Hierarchical Group

After establishing the PKI, we need to set up the system for SMPC (2). The consumer company of the group executes the SMPC setup step of SCALE-MAMBA, providing the specified parameters. As a result, it gets the data that will be used during the computation. We provided the details of the data generated in the SMPC setup step in the previous section (Section 5.3). After completing the SMPC Setup, the consumer company generates and compiles the SMPC program for the group computation. This program includes the details about the functions/computations being run, the data that needs to be kept secret (running computations on an encrypted version of data), and so on.

The last step before executing the group computation is the distribution of the data needed for the computation (3). In order to run a specific SMPC program in a group, companies in that group need to know the specified instructions and have the necessary data for the computation. The consumer company distributes all the needed data, including instructions, to the suppliers so that they can run the SMPC program. The required data contains the digital certificates of participants and the data files created during the SMPC setup. Also, in order to execute the SMPC program, companies receive the compiled SMPC program and related files from the consumer company.

After all the precursor steps are completed, the companies, including the consumer company, execute the SMPC program together (4). The steps outlined above are also visually represented in the architectural diagram shown in Figure 5.6. This diagram illustrates the computation process of a hierarchical group within our model's prototype. In SMPC, companies in the computation group jointly compute a function over their inputs; the function is run in a distributed way. Also, the function being computed is already compiled. Therefore, companies cannot make any changes to the computation program. Even if a company as an attacker modifies the program, it needs to recompile it and then distribute the compiled version to the other suppliers in order to run it successfully together. As we have authenticated channels in our approach, suppliers will know who is sending the computation program. Also, companies in the computation can see the function that will be computed. Therefore, they can identify the changes in the computation program.

The availability of the function being computed is required as companies would like to know what they will run on their systems. However, they cannot make any changes to the function, and neither can they see the data of the other participants. It is important to state that SMPC is a complex computation, and a large part of that complexity is because there is a lot of communication taking place. Therefore, executing the SMPC-program is the expensive part of the system: it costs a lot of time, storage, and resources. The function being computed also has an effect on the complexity of the computation. We will discuss in detail the issues related to system performance in the evaluation step of the prototype we implemented in Chapter 7.

The first three steps are steps that are run inside of an individual group. Each hierarchical group can run these steps independently from each other. However, in a business network, the computation results of the groups are dependent on each other; the computation result of one group can be used as an input into the computation of another group. Therefore, the run time of the computation of a group depends on the function being run and the relations with other groups in the business network.

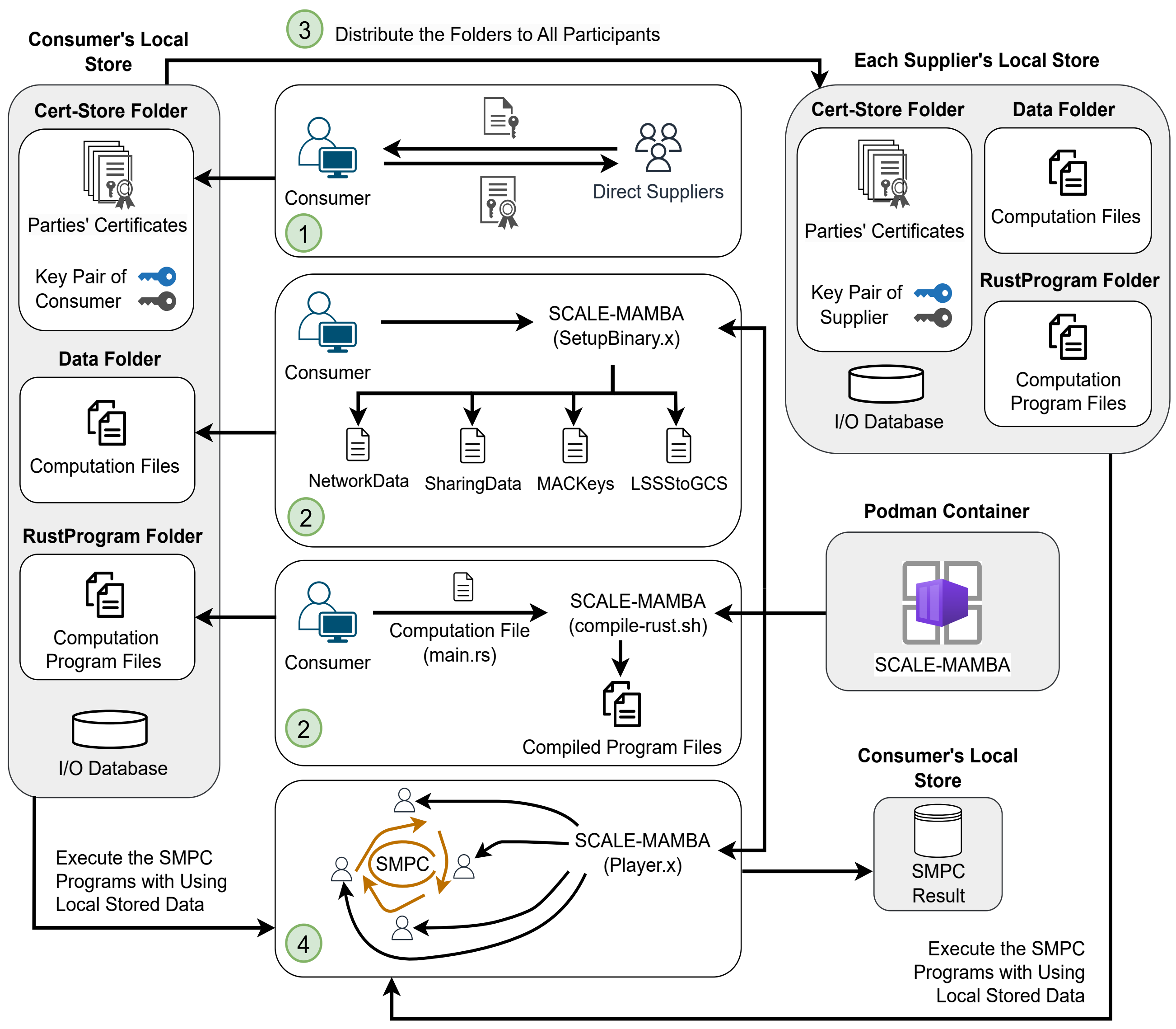


Figure 5.6: Architectural Diagram of the Prototype Model for a Single Hierarchical Group

5.5 Summary

In this chapter, we have seen how we implement our model. First, we have provided detailed information about the components that are needed to implement our model and the technical decisions that we made. Following that, we have described step-by-step how we implement our model to achieve security- and privacy-enhanced computations in hierarchical groups of a business network. In order to evaluate all aspects of our approach, from our threat modelling to our privacy-preserving model, and make an extensive assessment, we will next apply our approach to a case study. This will be the implementation of a life-cycle assessment.

6. Implementation of a Case Study: LCA

In this chapter, we implement a case study called life-cycle assessment (LCA) in our model to evaluate our approaches and perform a comprehensive assessment. In previous chapters, we also had a small running example, but we used that example as an illustrative case study to explain the concepts of our approaches, both the threat modelling approach and the scalable privacy-preserving model approach. The case study that will be presented in this chapter is a complex and non-trivial example that covers all the core concepts and aspects of our approaches in the previous chapters and is used for their evaluation. In this chapter, we first give brief background information about LCA – what it is and why it is important – and we demonstrate the traditional way of calculating an LCA. Then, we identify and enumerate potential security and privacy risks and threats in the traditional approach to LCA and derive the security and privacy requirements by following the steps of our threat modelling approach that we presented in Chapter 3. After that, we apply the scalable privacy-preserving model that we presented in Chapter 4 to traditional LCA to enhance the security and privacy of the LCA approach. As the last step, we make a comprehensive assessment of the enhanced LCA. We carry out a security and privacy analysis of the enhanced LCA. Then, in the next chapter, we will make a comprehensive performance evaluation of our privacy-preserving model approach in the context of LCA.

6.1 Life-Cycle Assessment (LCA)

Life Cycle Assessment (LCA) [19] is a standard method for assessing the ecological impact across the stages of a product, process, or service, guiding sustainable choices and improvements. The important point about LCA is that it evaluates the ecological sustainability of a product or a service in a quantitative way, and it requires exhaustive and comprehensible information about industrial activities to make an accurate evaluation. Therefore, LCA computations rely on life-cycle inventory (LCI) datasets, which are databases including essential information about the operation of a process or production step and the environmental emissions of industrial processes. This database is usually “static” and based on “historic” studies. As such, the result of an LCA is an approximation to the actual ecological impact. One reason for using a database instead of real-time data is the lack of the necessary infrastructure for collecting production data in real-time. The adoption of Industry 4.0 will solve this problem. Still, Industry 4.0 cannot solve the main reasons for *not* sharing data for LCA: the information that needs to be shared is of high business value and often considered confidential, and hence, companies are not willing to share this data [18, 19].

In many cases, companies and firms cannot operate independently. Hence, the business flow in the global and competitive environment is made via the supply chain [151]. A supply chain can include several *competitive companies* which are all operating the same industrial process, *entities* (suppliers, customers, factories, distributors, and retailers) which are operating the different steps of the same product, and an *aggregator* that gets the data from participants, aggregates it and shares the results with companies and customers [98]. The aggregator responsible for performing the computation can be either a trusted third party or a consumer company. *Traditional LCA* is a model calculated by a centralized system that takes inputs, known as *unit processes*, from each company in a supply chain. The unit processes include information about both the economic flows and the environmental flows within the supply chain. The economic flows describe the detailed supplier–consumer relationships (i.e. which company is ordering what quantities of a certain product), and the environmental flows describe in detail the flow from and into the global environment.

LCA does not necessarily have to include all the phases of a product's life cycle (cradle-to-grave). The extent of the life cycle considered in an LCA study can vary depending on the goals, scope, and resources available for the assessment. LCA studies can be conducted on different levels, known as "cradle-to-gate", "gate-to-gate", or "gate-to-grave" assessments [152]. In this section, we explain the calculation of a traditional LCA in detail by using a case study of LCA reported by the International Aluminium Institute [153, 154]. They calculated the LCA to specify resource consumption and important environmental aspects with regard to the worldwide production of primary aluminium. The purpose of this case study is to show that our approach can be applied to LCA. In our case study, we present a simplified illustration of LCA by focusing on the production process of aluminium, spanning from cradle-to-gate. However, it is important to note that extending the case study to cover the complete cradle-to-grave process would not enhance the goal of this case study, as it does not reveal new insights into its applicability. If we were to consider the entire life cycle of aluminium, including the disposal and recycling stages, the number of unit processes involved would significantly increase. Nevertheless, from a dependency perspective, the process remains the same: no new type of computation is introduced; instead, the matrices simply become larger.

Therefore, using a small model for the case study is a valid approach to demonstrate the applicability of our work to LCA. In Chapter 7, we will delve into the effects of the depth of a supply chain and the impact of adding more phases, participants or groups to it on the runtime of LCA.

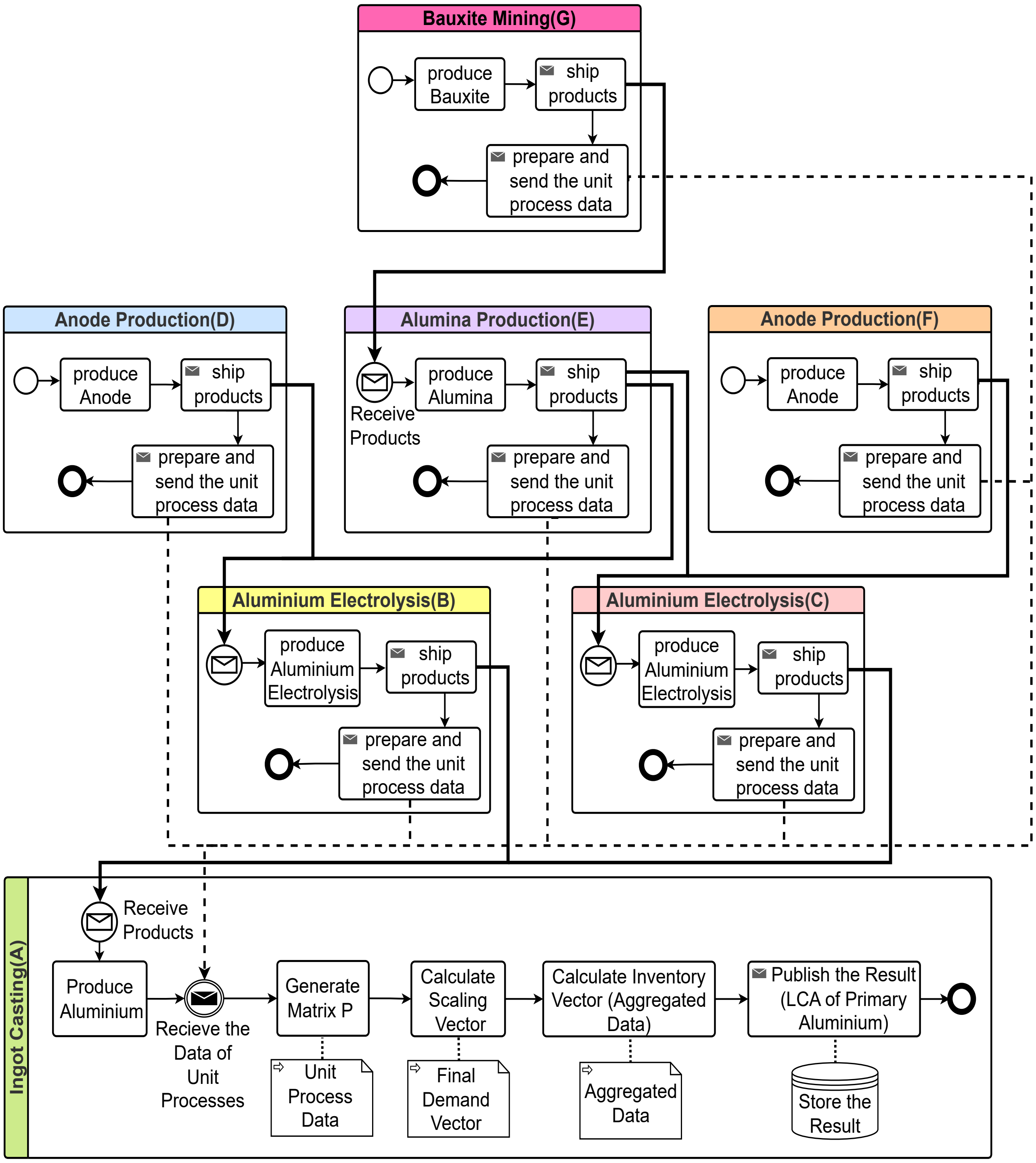
The life cycle of aluminium includes a number of supplier companies that provide unit processes (sub-materials), which are interconnected to allow calculations on the complete system for the production of aluminium. In Figure 6.1, we presented a BPMN diagram that illustrates the LCA of the aluminium production supply chain (inspired by [153, 154]) from mining bauxite to producing the actual aluminium ingots. The suppliers in this process, including companies that produce anodes and aluminium electrolysis, are potentially using different processes, having different ecological footprints. For the LCA of aluminium production, all suppliers in the supply chain send their unit process data to the main consumer company A. After receiving all unit process data, consumer company A does the actual LCA computation. In the following subsection, we explain how traditional LCA is calculated step-by-step using our case study example.

6.1.1 Calculation of Traditional LCA

In order to represent the structure of the supply chain and supplier–consumer relationships in the supply chain more clearly, we simplified Figure 6.1 to Figure 6.2. In this example, we focus on seven unit processes that are shown in bold rectangles in Figure 6.2: bauxite mining, alumina production, anode production (produced by two companies), aluminium electrolysis (produced by two companies), and ingot casting, respectively. The reason for focusing only on these seven unit processes is that the report [153, 154] provides the LCI datasets of these seven unit processes. The International Aluminium Institute stated that they did not add additional unit processes (energy production, transport, petrol coke, pitch production, etc.) to avoid non-elementary flows.

The production of each sub-material (produced by suppliers) and main material (aluminium) in the supply chain can be defined as unit processes, and each of them represents a simple dimensional matrix. All processes together represent an $(n+m)\times p$ matrix P that is generated by the consumer company (A) after receiving all unit process data from the supplier companies in the supply chain: p is the number of unit processes of the supply chain (P_0, P_1,...,P_p), and n and m are the number of economic and environmental flows, respectively. Every unit process may not include the same flow. In matrix P, the value of flows could be '0' if they are not included in a unit process.

$$P = \left(P_0 | P_1 | P_2 | \cdots | P_p \right) \tag{6.1}$$



Bauxite Mining(G)
produce Bauxite
ship products
prepare and send the unit process data
Anode Production(D)
produce Anode
ship products
prepare and send the unit process data
Alumina Production(E)
Receive Products
produce Alumina
ship products
prepare and send the unit process data
Anode Production(F)
produce Anode
ship products
prepare and send the unit process data
Aluminium Electrolysis(B)
produce Aluminium Electrolysis
ship products
prepare and send the unit process data
Aluminium Electrolysis(C)
produce Aluminium Electrolysis
ship products
prepare and send the unit process data
Ingot Casting(A)
Receive Products
Produce Aluminium
Recieve the Data of Unit Processes
Generate Matrix P
Unit Process Data
Calculate Scaling Vector
Final Demand Vector
Calculate Inventory Vector (Aggregated Data)
Aggregated Data
Publish the Result (LCA of Primary Aluminium)
Store the Result

Figure 6.1: BPMN Diagram of LCA of Aluminium Production

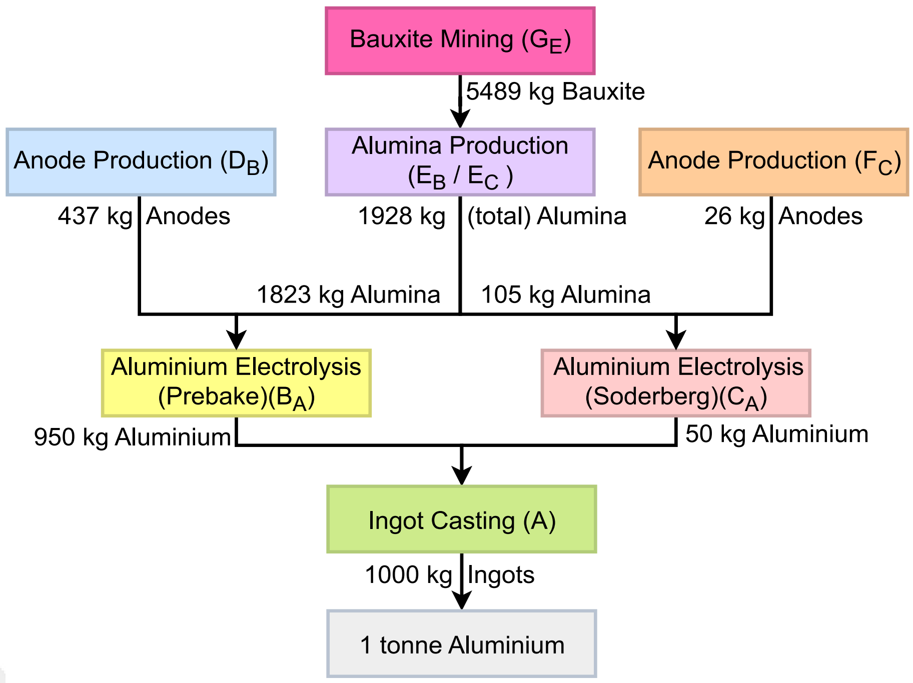


Figure 6.2: LCA of Aluminium Production Supply Chain

The process matrix P has two distinct parts: one includes matrix $\mathcal{A}$, which is an $n \times p$ matrix representing the flows within the economic system, and the other part contains matrix $\mathcal{B}$, which is an $m \times p$ matrix representing the flows from and into the environment (Equation 6.2).

$$P = \left(\frac{\mathcal{A}}{\mathcal{B}} \right) \tag{6.2}$$

In this example, we want to produce 1 000 kg of aluminium ingots (main product). As seen from Figure 6.2, electrolysis of 1 000 kg of alumina is used to produce 1 000 kg of aluminium (slab, billet, etc.). 950 kg of the aluminium is supplied by the *Prebake* facility and 50 kg of the aluminium is supplied by the *Soderberg* facility. The production of 1 000 kg of aluminium through this process results in the emission of 0.04 kg of particulates, 0.08 kg of NO_2, and 0.03 kg of SO_2 into the environment. This is one of the unit processes of the supply chain, P_0 (Equation 6.3). According to the *Prebake* company, 460 kg of anode and 1 919 kg of alumina are used to produce 1 000 kg of aluminium by electrolysis. In doing so, it emits 1.00 kg of particulates, 0.30 kg of NO_2, and 13.20 kg of SO_2 into the environment. This is another unit process of the supply chain, which we call P_1 (Equation 6.4).

$$\mathcal{A}_{P_0} = \begin{array}{c} \\ IC_A \\ AE_B \\ AE_C \end{array} \begin{array}{c} A \\ \left(\begin{array}{r} 1 \cdot 10^3 \\ -95 \cdot 10^1 \\ -5 \cdot 10^1 \end{array}\right) \end{array} \quad \mathcal{B}_{P_0} = \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \end{array} \begin{array}{c} A \\ \left(\begin{array}{r} 4 \cdot 10^{-2} \\ 8 \cdot 10^{-2} \\ 3 \cdot 10^{-2} \end{array}\right) \end{array} \quad P_0 = \begin{array}{c} \\ IC_A \\ AE_B \\ AE_C \\ P \\ NO_2 \\ SO_2 \end{array} \begin{array}{c} A \\ \left(\begin{array}{r} 1 \cdot 10^3 \\ -95 \cdot 10^1 \\ -5 \cdot 10^1 \\ 4 \cdot 10^{-2} \\ 8 \cdot 10^{-2} \\ 3 \cdot 10^{-2} \end{array}\right) \end{array} \tag{6.3}$$

$$\mathcal{A}_{P_1} = \begin{array}{c} \\ AE_B \\ AnP_D \\ AP_E \end{array} \begin{array}{c} B \\ \left(\begin{array}{r} 1 \cdot 10^3 \\ -46 \cdot 10^1 \\ -1\,919 \cdot 10^0 \end{array}\right) \end{array} \quad \mathcal{B}_{P_1} = \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \end{array} \begin{array}{c} B \\ \left(\begin{array}{r} 1 \cdot 10^0 \\ 3 \cdot 10^{-1} \\ 132 \cdot 10^{-1} \end{array}\right) \end{array} \quad P_1 = \begin{array}{c} \\ AE_B \\ AnP_D \\ AP_E \\ P \\ NO_2 \\ SO_2 \end{array} \begin{array}{c} B \\ \left(\begin{array}{r} 1 \cdot 10^3 \\ -46 \cdot 10^1 \\ -1\,919 \cdot 10^0 \\ 1 \cdot 10^0 \\ 3 \cdot 10^{-1} \\ 132 \cdot 10^{-1} \end{array}\right) \end{array} \tag{6.4}$$

In this case, the first column in matrix P (Equation 6.5) represents the unit process of ingot casting, while the second column shows the unit process of aluminium electrolysis production of the *Prebake* company.

$$P = (P_0 \mid P_1) = \begin{array}{c} \\ IC_A \\ AE_B \\ AE_C \\ AnP_D \\ AP_E \\ P \\ NO_2 \\ SO_2 \end{array} \begin{array}{c} \begin{array}{rr} A & B \end{array} \\ \left(\begin{array}{rr} 1 \cdot 10^3 & 0 \cdot 10^0 \\ -95 \cdot 10^1 & 1 \cdot 10^3 \\ -5 \cdot 10^1 & 0 \cdot 10^0 \\ 0 \cdot 10^0 & -46 \cdot 10^1 \\ 0 \cdot 10^0 & -1\,919 \cdot 10^0 \\ 4 \cdot 10^{-2} & 1 \cdot 10^0 \\ 8 \cdot 10^{-2} & 3 \cdot 10^{-1} \\ 3 \cdot 10^{-2} & 132 \cdot 10^{-1} \end{array}\right) \end{array} \tag{6.5}$$

In our example, we have seven unit processes and seven economic flows that are placed in matrix $\mathcal{A}$ (Equation 6.6), which is a 7×7 matrix relating products (in rows) to processes (in columns): ingot casting (1), aluminium electrolysis (2) (produced by two companies), anode production (2) (produced by two companies), alumina production (1), and bauxite mining (1), respectively. There are four environmental flows placed in matrix $\mathcal{B}$ (Equation 6.7), which is a 4×7 matrix relating processes (in columns) to environmental flows (in rows): particulates, NO_2, SO_2, and mercury (Hg), respectively.

$$\mathcal{A} = \begin{array}{c} \\ IC_A \\ AE_B \\ AE_C \\ AnP_D \\ AnP_F \\ AP_E \\ BM_G \end{array} \begin{array}{c} \begin{array}{rrrrrrr} A & B & C & D & F & E & G \end{array} \\ \left(\begin{array}{rrrrrrr} 1 \cdot 10^3 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 \\ -95 \cdot 10^1 & 1 \cdot 10^3 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 \\ -5 \cdot 10^1 & 0 \cdot 10^0 & 1 \cdot 10^3 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 \\ 0 \cdot 10^0 & -46 \cdot 10^1 & 0 \cdot 10^0 & 1 \cdot 10^3 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 \\ 0 \cdot 10^0 & 0 \cdot 10^0 & -515 \cdot 10^0 & 0 \cdot 10^0 & 1 \cdot 10^3 & 0 \cdot 10^0 & 0 \cdot 10^0 \\ 0 \cdot 10^0 & -1\,919 \cdot 10^0 & -2\,105 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 1 \cdot 10^3 & 0 \cdot 10^0 \\ 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & 0 \cdot 10^0 & -2\,847 \cdot 10^0 & 1 \cdot 10^3 \end{array}\right) \end{array} \tag{6.6}$$

$$\mathcal{B} = \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \\ Hg \end{array} \begin{array}{c} \begin{array}{ccccccc} A & B & C & D & F & E & G \end{array} \\ \left(\begin{array}{rrrrrrr} 4 \cdot 10^{-2} & 1 \cdot 10^{0} & 2 \cdot 10^{0} & 2 \cdot 10^{-1} & 0 \cdot 10^{0} & 5 \cdot 10^{-1} & 1 \cdot 10^{-1} \\ 8 \cdot 10^{-2} & 3 \cdot 10^{-1} & 4 \cdot 10^{-1} & 5 \cdot 10^{-1} & 1 \cdot 10^{-2} & 6 \cdot 10^{-1} & 0 \cdot 10^{0} \\ 3 \cdot 10^{-2} & 132 \cdot 10^{-1} & 8 \cdot 10^{0} & 43 \cdot 10^{-1} & 1 \cdot 10^{-1} & 11 \cdot 10^{-1} & 0 \cdot 10^{0} \\ 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 12 \cdot 10^{-2} & 0 \cdot 10^{0} \end{array} \right) \end{array} \tag{6.7}$$

As the next step, in order to make the life cycle assessment of the given aluminium product supply chain, we define the goal. The goal includes the specification of the required performance of the system. For this, we determine a *reference flow*. The reference flow is referred to as the final demand vector of the system, and it represents the quantity of material(s) produced by the supply chain. In our case, 1 000 kg of primary aluminium (ingot casting) is taken as a reference flow. Even though only one of the economic flows is the reference flow, we specify the complete set of these flows, which is vector f. It is referred to as the *final demand vector*. As a final aspect, we obtain the set of all environmental flows connected with the reference flow under consideration. This set is called an *inventory vector* g. The aggregated flows of the entire system are placed in the final demand vector and the inventory vector [155]. The system vector (q) is:

$$q = \left(\frac{f}{g} \right) \quad f = \begin{array}{c} \\ IC_A \\ AE_B \\ AE_C \\ AnP_D \\ AnP_F \\ AP_E \\ BM_G \end{array} \begin{array}{c} A \\ \left(\begin{array}{c} 1 \cdot 10^{3} \\ 0 \cdot 10^{0} \\ 0 \cdot 10^{0} \\ 0 \cdot 10^{0} \\ 0 \cdot 10^{0} \\ 0 \cdot 10^{0} \\ 0 \cdot 10^{0} \end{array} \right) \end{array} \quad g = \begin{pmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{pmatrix} \tag{6.8}$$

To calculate the aggregated data of environmental flows (g_1, g_2, g_3, and g_4) in the system, it is necessary to calculate the *scaling vector*, which is a $p \times 1$ matrix (i.e. a vector) representing scaling factors for the unit processes; in our case, it is a 7×1 matrix. The scaling vector is a key component of the LCA calculation, as it is used to weight the impact of the different inputs and outputs in the supply chain. The scaling vector s:

$$s = \mathcal{A}^{-1} \cdot f = \left(\begin{array}{r} 1 \cdot 10^{0} \\ 95 \cdot 10^{-2} \\ 5 \cdot 10^{-2} \\ 437 \cdot 10^{-3} \\ 2\,575 \cdot 10^{-5} \\ 19\,283 \cdot 10^{-4} \\ 54\,899 \cdot 10^{-4} \end{array} \right) \tag{6.9}$$

After calculating the scaling vector for the supply chain, we calculate the vector of the system-wide aggregated environmental flows. The *inventory vector* g:

$$
g = \mathcal{B} \cdot s
$$

$$
= \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \\ Hg \end{array}
\begin{array}{c}
\begin{array}{ccccccc} A & B & C & D & F & E & G \end{array} \\
\left(\begin{array}{rrrrrrr}
4 \cdot 10^{-2} & 1 \cdot 10^{0} & 2 \cdot 10^{0} & 2 \cdot 10^{-1} & 0 \cdot 10^{0} & 5 \cdot 10^{-1} & 1 \cdot 10^{-1} \\
8 \cdot 10^{-2} & 3 \cdot 10^{-1} & 4 \cdot 10^{-1} & 5 \cdot 10^{-1} & 1 \cdot 10^{-2} & 6 \cdot 10^{-1} & 0 \cdot 10^{0} \\
3 \cdot 10^{-2} & 132 \cdot 10^{-1} & 8 \cdot 10^{0} & 43 \cdot 10^{-1} & 1 \cdot 10^{-1} & 11 \cdot 10^{-1} & 0 \cdot 10^{0} \\
0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 12 \cdot 10^{-2} & 0 \cdot 10^{0}
\end{array} \right)
\end{array}
\cdot
\left(\begin{array}{r}
1 \cdot 10^{0} \\ 95 \cdot 10^{-2} \\ 5 \cdot 10^{-2} \\ 437 \cdot 10^{-3} \\ 2\,575 \cdot 10^{-5} \\ 19\,283 \cdot 10^{-4} \\ 54\,899 \cdot 10^{-4}
\end{array} \right)
$$

$$
= \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array}
\left(\begin{array}{r} 2.6905 \\ 1.7605 \\ 16.9728 \\ 0.2314 \end{array} \right)
\tag{6.10}
$$

$$
= \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \\ Hg \end{array}
\begin{array}{c}
\begin{array}{ccccccc} A & B & C & D & F & E & G \end{array} \\
\left(\begin{array}{rrrrrrr}
4 \cdot 10^{-2} & 1 \cdot 10^{0} & 2 \cdot 10^{0} & 2 \cdot 10^{-1} & 0 \cdot 10^{0} & 5 \cdot 10^{-1} & 1 \cdot 10^{-1} \\
8 \cdot 10^{-2} & 3 \cdot 10^{-1} & 4 \cdot 10^{-1} & 5 \cdot 10^{-1} & 1 \cdot 10^{-2} & 6 \cdot 10^{-1} & 0 \cdot 10^{0} \\
3 \cdot 10^{-2} & 132 \cdot 10^{-1} & 8 \cdot 10^{0} & 43 \cdot 10^{-1} & 1 \cdot 10^{-1} & 11 \cdot 10^{-1} & 0 \cdot 10^{0} \\
0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 0 \cdot 10^{0} & 12 \cdot 10^{-2} & 0 \cdot 10^{0}
\end{array} \right)
\end{array}
\cdot
\left(\begin{array}{r}
\boxed{1 \cdot 10^{0}} \\ \boxed{95 \cdot 10^{-2}} \\ \boxed{5 \cdot 10^{-2}} \\ \boxed{437 \cdot 10^{-3}} \\ \boxed{2\,575 \cdot 10^{-5}} \\ \boxed{19\,283 \cdot 10^{-4}} \\ \boxed{54\,899 \cdot 10^{-4}}
\end{array} \right)
\tag{6.11}
$$

In total, 1000 kg of aluminium production emits 2.69 kg of particulates, 1.76 kg of NO_2, 5.69 kg of SO_2, and 0.23 g of mercury (Hg) into the environment. The calculation of the system's inventory vector (g) involves active participation from all companies within the supply chain. We have included an annotated (coloured) equation, which is 6.11, to visually demonstrate the involvement of all companies in the operations. All companies in the supply chain contribute to the system's inventory vector (g) calculation by sending their unit process data, economic and environmental flows, to the main consumer, referred to as A. As demonstrated, the values in each column of matrix $\mathcal{B}$ are obtained from the participating companies, while the scaling factors for each company are represented by the corresponding elements (rows) in the scaling vector s. For more details about the calculations and the data in the matrices, we refer the reader to [154, 155, 156].

If we naively turn to the system using a method that secures the computation, such as *secure multi-party computation*, it would result in a system with numerous participants and complex operations, which would be extremely slow. To address the challenges of shared data security and privacy along with system performance, we will apply our scalable privacy-preserving approach. However, before implementing this approach, we will do a thorough threat modelling for the case study using our established threat modelling approach.

6.2 Modelling of Life-Cycle Assessment Network

To accurately evaluate the risks of an inter-organizational collaboration model, it is essential to first model the network itself. This will enable a thorough assessment of the risks involved. In the previous section, we gave a brief overview of LCA and showed how to calculate LCA for aluminium production. We used a BPMN diagram to represent the business processes in the LCA model. In this section, we will delve deeper into modelling the LCA supply chain of aluminium production by identifying key considerations. We will utilize our threat modelling, as shown in Chapter 3, to help identify these matters.

Before we start threat modelling for the LCA supply chain of aluminium production, we need to set the assumptions. In this example, the business goals are to produce aluminium and conduct a life-cycle assessment of the aluminium production supply chain. To calculate the LCA of the aluminium production supply chain, participating companies provide their input data (economic and environmental flows) to the main consumer company, Company A. The supply chain network information is also available to all participating companies, which allows them to see the other companies in the supply chain and their relationships with one another. In addition, the result of the calculation is shared with all participating companies, potentially enabling them to access the trade secrets of others. Furthermore, consumer companies and their suppliers have direct relationships, enabling consumer companies to deduce the environmental impact of their suppliers.

6.2.1 Identifying the Participants and Their Roles

In this LCA collaboration, there are seven companies. One main company (the end consumer), which is A, is responsible for collecting the data from the companies and performing the aluminium production LCA. It plays only the *consumer* role. All the other companies are in a *supplier* role, providing their unit process data to A. In the supply chain of aluminium production, most companies also have additional roles. First of all, all companies except A have a supplier role, by which they supply their product to their consumer in the supply chain. A, B, C, and E have a consumer role, by which they consume the material they received from their suppliers. In addition to this, all companies can be in a *complementor* role for the supply chain. For example, company E has a *complementor* role for both D and F; they provide anode and alumina production, respectively, to the aluminium electrolysis process. B and C are in a *competitor* role with each other, while they have a *complementor* role for the supply chain. Similarly, D and F are in a *competitor* role with each other, while they have a *complementor* role for the supply chain. The multiple roles are as follows:

- *Trusted third-party*: None
- *Consumer*: A, B, C, and E
- *Supplier*: B, C, D, E, F, and G
- *Competitor*: B and C; D, and F
- *Complementor*: All companies

6.2.2 Identifying the Collaborations

The collaboration we have in the aluminium production supply chain contains companies with various roles: competitor, complementor, supplier, and consumer. Therefore, the collaboration is a *network collaboration* where a group of companies works together with the shared goals of producing aluminium and calculating the LCA of aluminium production. In our scenario, we assume a centralized collaboration, in which A also acts as the trusted main consumer company for the LCA computation. This means that all other companies are sending their data to A, even though only its direct suppliers (B and C) have a contractual relationship with A, while the indirect suppliers – D, E, F, and G – do not.

6.2.3 Identifying the Assets

In an LCA calculation, the main components are the *economic flows* and *environmental flows*. Economic flows include data on the materials required for production, such as the amounts of resources used by the company. Environmental flows include data on emissions into the environment as a result of the production process. For example, according to the *Prebake* company, to produce 1 000 kg of aluminium electrolysis, 460 kg of anode and 1 919 kg of alumina are used. These are economic flows of the *Prebake* company (B). In doing so, it emits 1.00 kg of particulates, 0.30 kg of NO_2, and 13.20 kg of SO_2 into the environment. These are environmental flows of the *Prebake* company. Together, these flows provide a comprehensive understanding of the environmental and economic impact of a product or process. Therefore, *economic flows* and *environmental flows* are considered to be assets.

Companies' relationships are confidential, so they are also considered to be assets. For example, companies do not want to share the names of their own suppliers with their customers. Another example from our case study is that *Anode Production* company F does not want other competitive companies like D to know their relationship with their consumer company, *Aluminium Electrolysis* (C). In addition, companies can be sensitive about revealing their identity to the whole supply chain, so the *membership* information is also considered confidential.

6.3 Modelling the Attack and Threat Landscape

Companies at different levels of a supply chain can be grouped into particular risk forms. When we categorize them, we consider the type and sensitivity of the information being processed, the threat actors, the capability of likely threats made by the threat actors, the quality of service ensured, and so on. These can help to control and manage supply chain systems and reduce the amount of revealed confidential data in an LCA. In the previous section, we categorized and identified the information being processed. In this section, we identify threat actors and determine the potential threats made by threat actors to the identified assets.

6.3.1 Identifying the Threat Actors

Our main aim of threat modelling the LCA collaboration is to design an inter-organization software system that provides better security than the traditional approach of LCA. In particular, our aim is to ensure that companies only need to share the information that is strictly necessary for the LCA computation and only with those companies that need to have access to this information. Thus, we focused on insider attacks – in other words, internal threat actors. In our case study example:

- *Ingot Casting* company (A) can act as a *downstream threat actor* as it is responsible for collecting data from participating companies in the supply chain and computing the LCA for the supply chain (the traditional LCA model).
- All other companies can be *upstream threat actors*; they have a supplier role that supplies the needed data to the consumer (A).
- Moreover, all companies except A can also act as *lateral threat actors*. The supply chain includes competitive and complementor companies, and a company as an insider attacker may want to attack its competitive or complementor companies.

6.3.2 Identifying the Threats and Mapping Them with Actors and Assets

As we identified above, there are three types of insider threat actors in the LCA model. In the following, we identify the threats posed by those threat actors and map them against the identified assets in the LCA model. In identifying the threats, we apply our threat modelling approach from Chapter 3. As identified and analysed in Section 3.3, there can be various different types of threats in collaborative business networks. In our LCA case study, we focus more on the threat type that our approach makes a difference to, which relates to the confidentiality of the data shared in an LCA computation. Therefore, in the following, we identify the threats to the traditional approach of LCA based on our focus. However, we also identify the other types of threats briefly. In the following, we cover these threats by threat actor type.

Attacks from Downstream Threat Actors

In the traditional approach to LCA, an LCA computation is performed by the consumer company that is the last link in the supply chain system or by a third party that makes the computation on behalf of the supply chain. In our approach, as we are interested in insider attacks, we focus on attacks made by consumer companies when they are in charge of making the LCA computation (presented in Section 6.1). In the traditional approach of LCA, a consumer company as an aggregator gets inputs from the participating companies of the LCA supply chain, and these inputs include confidential information of the companies (assets of the companies). In the following, we identify the attacks that can be made by a downstream threat actor, the *Ingot Casting* company A, on the confidentiality of the identified assets in our case study (Figure 6.2):

- *Competitive/Complementor Sub-suppliers Membership*: Company A gets information about who the competitive/complementor sub-suppliers of its suppliers are. As an attacker, A could disclose this information to switch them or make a special contract with them; controlling the memberships of the supply chain. For example, if A wants to change supplier company F, supplying anode production, it can share the information with *Aluminium Electrolysis* company C that company B, which is the competitive company of C, gets anode production from supplier company D. C can make a deal with D to supply anode production and take out F from the supply chain.
- *Business Relationships of Suppliers*: Consumer company A can easily exploit the information about who communicates with whom to check and conduct the relationships in the supply chain. For example, A can learn who supplies *alumina* material to *Aluminium Electrolysis* companies B and C and infer the trade secrets of those companies.
- *Production Processes of Suppliers*: A consumer company could be interested in knowing the manufacturing details of suppliers, such as how much production is produced by a supplier, in order to negotiate with the supplier. Company A could know how much *electrolysis* supplier C produces and introduces into the market. For a consumer company, it is easy to get this information because this data (economic flows) is sent from supplier companies to consumer companies for the traditional LCA computation.
- *Environmental Flows of Suppliers*: The consumer company may want to attack the environmental flow data of its suppliers to evaluate the environmental ethics of suppliers. For example, suppliers B, C, D, E, and F emit NO_2 and SO_2 into the environment (we refer the reader to [153, 154] for the detailed information). These emissions are important for life-cycle assessment, and they are available to consumer company A in the traditional LCA computation.

Not only information disclosure, but also other types of attacks can be carried out by a downstream threat actor:

- *Spoofing*: A consumer company, as a downstream attacker, may want to pretend to be somebody else. For example, if there is a lack of authentication in the system, company A might be able to convince *Alumina Production* company E that they are actually *Aluminium Electrolysis* company B. This could allow A to learn trade secrets between B and E that A might use to influence its offer to B.
- *Tampering*: If consumer company A is in charge of making the LCA computation, it can easily modify the data of companies – the *economic flows* and the *environmental flows* – provided for the LCA computation; it gets all inputs as clear data (no encryption) from participating companies in the supply chain. This could affect the result of the LCA computation and also the assets of the companies in the LCA supply chain, such as their reputation, knowledge of their production processes, etc.
- *Denial of Service*: If company A has the authority to manage the system, it can apply a denial of service attack to any company in the LCA supply chain. For example, consumer company A might want to change one of the sub-supplier companies (e.g. F) in the supply chain for some financial reasons or because of the deal it made with another competitive company (e.g. D). A can mount a denial of service attack on sub-supplier F to prevent it from accessing the service and destroying its business relationship with C.
- *Elevation of Privilege*: Company A, in the traditional LCA case study, has the right and permission to get the inputs (*environmental and economic flows*) from the suppliers of the LCA supply chain. Having these specific rights and privileges can cause the disclosure of confidential information, including the assets of companies in the LCA.
- *Linkability, Identifiability, Detectability and Non-repudiation*: Since the main consumer company has the inputs from the participating companies of the LCA supply chain, as a downstream threat actor, it can identify and link them, detect other information, and disclose any information that it would like to know, and the companies cannot deny or refuse this evidence. For example, company A learns information about the business relationships and production processes of companies from the inputs (economic flows) it gets from companies. A can identify that *Alumina Production* company E has two business relationships in the supply chain, and it can link that E supplies alumina production to two competitive companies. Also, E cannot deny this information, as this information is also provided to the consumer company by these companies.

Attacks from Upstream Threat Actors

Some information provided to the LCA computation is considered secret for companies because they can disclose some confidential data, such as production processes (quantity of materials, emissions, etc.), business processes, trade secrets, and so on. A supplier who acts as an attacker may seek to obtain this information for the purpose of increasing their profits. They may use this information to determine the pricing of their product supplied to the consumer company, manipulate the pricing of the supplied product, or analyze market trends. Also, it is possible for a supplier to collude in the supply chain to evaluate the supply chain and use this information for their benefit. A supplier company as an upstream threat actor can carry out attacks to learn or disclose four types of information about the consumer company in the supply chain:

- *Consumer Membership*: In some supply chain systems, the consumer company does not alter or integrate the product with something else; it can be in a distributor role. Therefore, a supplier company can attack the system to learn who the consumer company is.
- *Business Relationships of Consumer*: A supplier might want to attack the system to know with which companies the consumer company has a relationship in the supply chain. The supplier can use this information to make secret deals and agreements.
- *Production Process of Consumer*: An attacker in a supplier role can collude with other suppliers to get the confidential data of the consumer company. In our case study, for example, if *Aluminium Electrolysis* suppliers B and C collude in the supply chain, they can easily infer the economic flows data of their consumer company A. They can use this information to get special offers from the consumer company or make new deals. Also, in the example, supplier E, *Alumina Production* company, sends its input related to the production of *alumina* to consumer company A for the LCA computation. When A shares the total result of the supply chain and if supplier E realizes that it is the only supplier for a certain material (*alumina*), it can disclose some confidential data of its consumer companies (B and C), such as trade secrets (the quantity of the product, the income of the consumer company, and so on).
- *Environmental Flows of Consumer*: Not only economic flows but also environmental flows are essential in LCA. Therefore, a supplier company can attack the system to get the environmental flows of the consumer company. If a supplier is able to discover which environmental flows are emitted by which emission to the environment, it can carry out an attack to get data about the environmental flows of the consumer company.

Also, a supplier, as an upstream threat actor, can attempt to perform other types of attacks:

- *Spoofing*: A supplier might want to impersonate a consumer company to learn information that is not available to them. For example, if *Anode Production* company D can impersonate A, it can convince its competitor F to reveal the business relationships and production processes between F and its consumer C, *Aluminium Electrolysis*. D can use this information to trade with and build relationships with C and destroy F's relationship with C.
- *Tampering*: If a supplier, as an upstream threat actor, is able to modify the information that A has or obtains to calculate the LCA computation, such as increasing the NO_2 emission value of the product (environmental flows) in the calculation, it can destroy the reputation and business relationships of A because of its environmental impacts.
- *Denial of Service*: A denial of service attack on consumer company A might destroy the whole supply chain. A may not provide the services (LCA computation) to companies, and this can cause some problems like delays in processes, miscommunication between companies, etc.
- *Elevation of Privilege*: If a supplier company becomes a user that has authenticated access to another company's system and is able, e.g., by exploiting a vulnerability in a software system, to upgrade their access to one with more privileges, they can access and learn all the inputs provided by companies in the supply chain for the LCA computation of the supply chain.
- *Linkability, Identifiability and Detectability*: When the result of the LCA computation is released, a supplier company may infer some confidential information about the consumer company by using its own input and the result. Company B, for example, can deduce from the findings that it is not the only one that supplies aluminium electrolysis to consumer company A. B can detect company C from the LCA result as it knows who joined the supply chain. Moreover, B can link C's business relationships in the supply chain.

Attacks from Lateral Threat Actors

A supplier company as a lateral threat actor can also want to know the confidential data of other suppliers in the LCA supply chain to analyse the market and gain/increase its profit. It can use this information to negotiate with other suppliers or the consumer company and compete on price:

- *Competitive/Complementor Supplier Membership*: A supplier, as a lateral threat actor, could want to know whether there are any competitive suppliers in the supply chain. Our example (Figure 6.2) includes competitive suppliers like B and C that supply the same service (*Aluminium Electrolysis*) to consumer company A. Also, a supplier might want to attack information about who the suppliers are that supply other materials to the consumer company to make a deal or special agreement without informing the consumer company. These data disclosures can cause some serious problems for companies, like revealing secret agreements or deals.

- *Sub-suppliers' Membership of Other Suppliers*: An attacker (e.g. C) might be interested in learning the relationship between another supplier (B) and its sub-suppliers (E).
- *Business Relationships of Other Suppliers*: The materials of some suppliers are strongly related to each other, such as *alumina* (E) and *aluminium electrolysis* (B and C). Therefore, for example, supplier B, as a lateral threat actor, could want to know the business relationships of E to make a special deal with E. On the other hand, B can negotiate with C to agree on the price that they offer to E.
- *Production Processes of Other Suppliers*: A supplier (e.g. F) might want to attack to get data about how much/many material(s) are produced and supplied by other suppliers in the supply chain. F, for example, is not the only supplier that supplies *Anodes*; F supplies 58 kg of Anodes to C, but the total amount required by C is 440 kg (Figure 6.2).
- *Environmental Flows of Other Suppliers*: A supplier may want to attack data about the environmental flows of other suppliers (they can be competitive or related companies) to analyse its data and use this information to increase its reputation in the market. Also, knowing others' environmental flows can help the supplier to improve its product and increase sales.

A company, as a lateral threat actor, can also perform other different types of attacks on other companies in the LCA supply chain for different aims:

- *Spoofing*: A supplier company might want to impersonate its competitive company in the supply chain. Therefore, for example, if the system has insufficient authentication, *Aluminium Electrolysis* company C, as a lateral threat actor, might be able to convince A that they are actually their competitor company B. This could allow C to submit incorrect environmental flows data to A that A believes to be from B. This could damage the reputation of B. As a consequence, C might be able to take over the supplier relationship to A.
- *Tampering*: A supplier, as a lateral threat actor, can attempt to modify the data of a competitive company to destroy its reputation or business relationships in the supply chain.
- *Denial of Service*: If a supplier is not ready to supply the needed data for the computation, it might want to prevent other suppliers from accessing the service. This action has no direct effect on assets, but it would cause some problems, such as delays in processes and miscommunication between companies.

- *Elevation of Privilege*: A supplier might be interested in gaining rights to access information about other suppliers (competitive or complementary) to learn about their business processes with the consumer or the production processes. For example, *Anode Production* company D, as a lateral threat actor, could attack *Bauxite Mining* company E to learn about potential consumer companies like C, *Aluminium Electrolysis* company.
- *Linkability, Identifiability and Detectability*: A supplier can detect, identify, and link some information in a supply chain if there is a vulnerability in the chain. For example, if company D knows all the participants in the supply chain, D can identify what products they supply to the supply chain. Then D can detect that E is the only company supplying *alumina*. D can deduce from the result of the LCA computation that all *mercury* (Hg) emission belongs to company E.

6.3.3 Deriving Security and Privacy Requirements

From our threat model, we derived the security and privacy requirements for the LCA computation of a supply chain. In particular, we derived the requirements for ensuring the confidentiality and privacy of shared data in the LCA supply chain that contains and/or affects the assets of companies. Firstly, membership information of companies in a supply chain can be considered confidential due to the possibility of revealing some private information about companies, such as their identity. Therefore, the membership information of any company in a supply chain should be kept secret. Suppliers in a supply chain should not be aware of each other if they do not have a direct relationship with each other. Not only suppliers but also consumer companies should be limited in knowing the members of the supply chain. A consumer company can know its direct suppliers, but it should not know any supplier that it does not have a direct relationship with. Thus, a consumer company should not know the membership information of sub-suppliers in the supply chain.

Economic flows (business relationships and production processes) are also important and confidential information in supply chains. Suppliers may know how many participants are in the computation. However, they should not know who the other suppliers are, what they supply to the consumer company, or the other companies in the supply chain and how much/many material(s) they supply. They should also not know who the sub-suppliers of other suppliers are and what and how much/many material(s) these sub-suppliers supply to other suppliers. Furthermore, a consumer company should not know whether its suppliers have any supplier companies or who they are, and/or how many there are. Also, it should not learn how much/many material(s) its suppliers get from their sub-suppliers and what the materials are.

Not only economic flows but also environmental flows need to be kept secret. Suppliers in a supply chain may know the list of environmental flows in the computation, but they should not know the value of environmental flows of other suppliers or sub-suppliers, or the consumer company. In addition, although a consumer company can specify the environmental flows in the computation, it should not learn the individual data sets (the estimation of environmental flows) of its direct suppliers or indirect suppliers in the supply chain.

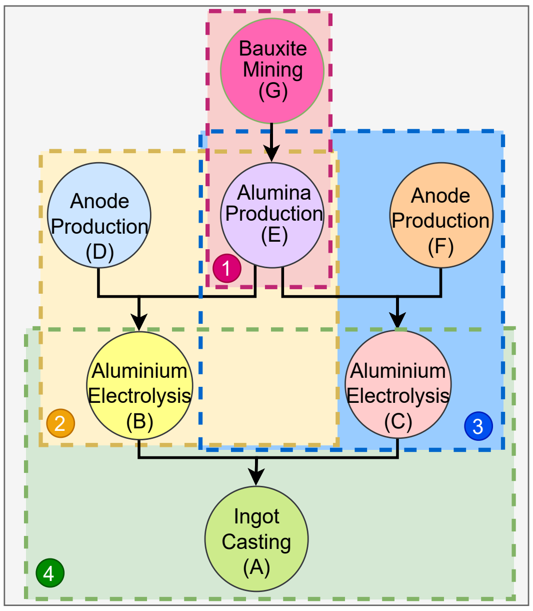
As is stated in Subsection 3.3.4, in an ideal system, all companies in a supply chain should be aware of the companies with which they have direct relationships, and they should just learn the information (the total result) that they could use to analyse the market, evaluate their own profits, and produce a more sustainable product/system.

6.4 Confidentiality-Enhanced Life-Cycle Assessment

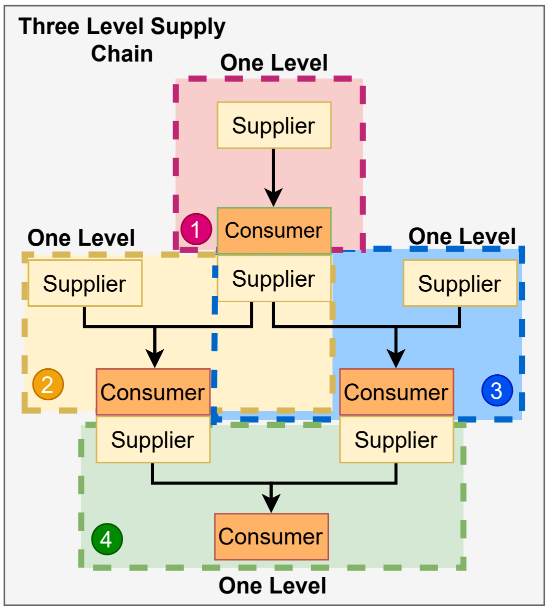
As we have seen from previous sections, LCA can reveal confidential data within a supply chain. Consequently, many companies are reluctant to share data that is required for close supply chain collaboration in general (see, e.g., [97]) or, in particular, LCA (see, e.g., [98]). To overcome this challenge, we apply our novel approach that we presented in Chapter 4 to LCA; our approach uses SMPC [21] to provide increased protection of the data that is required for completing an LCA. Compared to naïvely applying SMPC to the computed Equation 6.11 jointly, our approach provides additional security properties and, in most cases, also performance improvements (the details will be provided in the following sections).

6.4.1 Confidentiality-Enhanced LCA Model

Hierarchical Grouping. Recall our example from Section 6.1 (Figure 6.2): seven companies (called A–G) form the supply chain for aluminium ingots. They provide the following unit processes: Bauxite Mining (G), Anode Production (D), Aluminium Electrolysis (B), Alumina Production (E), Aluminium Electrolysis (C), Anode Production (F), and Ingot Casting (A). This supply chain can naturally be divided into a hierarchy based on "direct supplier relationships" (see Figure 6.3a).



(a) Hierarchic LCA Groups



(b) Roles of Parties in Groups

Figure 6.3: Hierarchical Grouping of Aluminium Production Supply Chain

Each group of companies (e.g. group 2 in Figure 6.3a) has one consumer (company B for group 2) and several direct suppliers (D and E for group 2). Within a large supply chain, an individual company can be both a supplier and a consumer. For example, company B is participating as a supplier in group 4 and as a consumer in group 2 (see Figure 6.3b). The core idea of our approach is to carry out local computations, in this case local LCAs, for each group using SMPC. This also requires a secure communication infrastructure for each group. In the following, we discuss these aspects in more detail.

Creating a Secure Communication Infrastructure. Before making computations, we should ensure secure communications between participants. To create secure communication channels, we use public-key infrastructures (PKIs) using X.509 certificates, where the consumer company takes over the role of the certificate authority (CA) for the group. Suppliers use pseudonyms when joining a PKI of a local LCA group, and, by default, we also recommend using different pseudonyms when joining different supply chains or if a company participates in different levels or groups of the same supply chain. This ensures that only the consumer company knows its own suppliers, but a supplier cannot learn the real identity of the other suppliers within the same group. Figure 6.4 demonstrates the PKI of the aluminium production supply chain. While the final setup (see Figure 6.4) looks like a PKI with hierarchical CAs, there is an important difference: the CAs are independent of each other, i.e. there is no common root CA for the complete supply chain. However, it is important to state that there is a trusted public CA for authentication of group CAs (details are explained in Section 5.2).

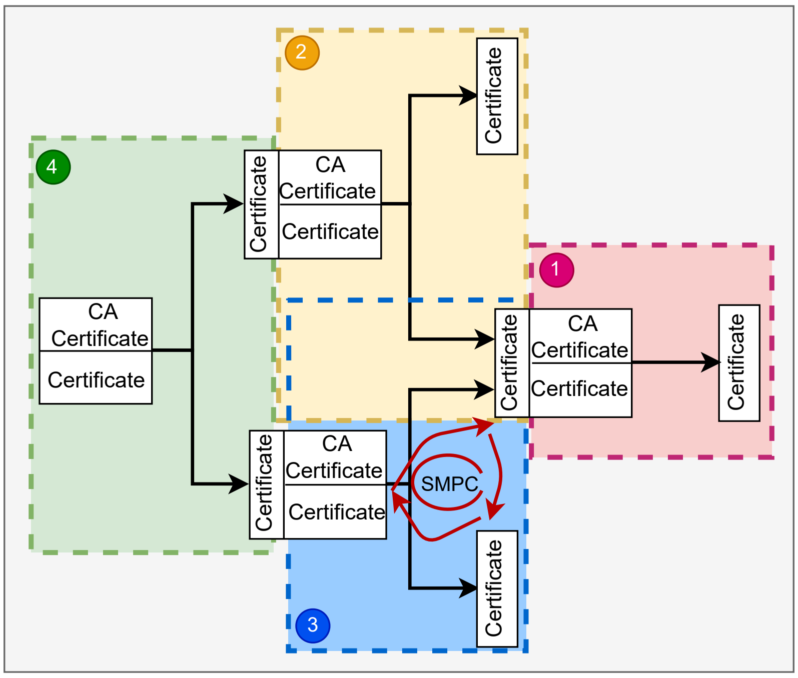


Figure 6.4: Distribution of Certificates in Aluminium Production Supply Chain

LCA Using SMPC. To protect the confidential information that companies provide as part of an LCA, we use SMPC. The fundamental security property of SMPC is that all participants only know their own input into the joint computation and, if published, the final output. As SMPC requires that all partners of the joint communication exchange messages with each other, each member of a group can learn the size of the group and the pseudonyms used for creating the X.509 certificates.

We compute the environmental impact for one unit of production (e.g. 1 piece or 1 kg of the produced product). This allows us to simplify the joint LCA computation within one group i to

$$s_i = \mathcal{A}_i^{-1} \cdot f_i \qquad\qquad g_i = \mathcal{B}_i \cdot s_i \tag{6.12}$$

where $\mathcal{A}_i$ is the $n \times p_i$ matrix representing the economic flows of participants within group i, $\mathcal{B}_i$ is the $m \times p_i$ matrix representing the environmental flows of participants within group i, f_i is a $p_i \times 1$ normalized final demand vector determined by the consumer company of group i, and s_i is a $p_i \times 1$ normalized scaling vector (scaling factors for each company in a computation group) computed by the consumer company of group i. As s_i can be computed by the consumer company of group i without further input from its suppliers, its computation does *not* require the application of SMPC. Also, as we compute the environmental impact for one unit of production, the final demand vector will not have an effect on the scaling vector; the scaling factors of companies in scaling vector s will have the same coefficients as the economic flows of the consumer company. This significantly reduces the number of operations that require SMPC and improves the data confidentiality of the economic flows. In most real-world scenarios, the sizes of these groups will be rather small compared to the overall number of participants in the supply chain (i.e. $p_i \ll p$). Therefore, we expect the computation for one computation group to be significantly faster than an SMPC-based LCA for the whole supply chain. Moreover, independent groups (in our example shown in Figure 6.3a, group 2 consisting of the companies B, D, and E and group 3 consisting of the companies C, F, and E) can perform the LCA computation of their group in parallel, resulting in a further performance gain (speedup).

Putting Everything Together. We assume that LCA is initiated by the main consumer company, e.g. the company producing the final product or the company recycling the final product. In our example, the main consumer company is company A, producing aluminium ingots. The consumer company contacts "down-stream" its direct suppliers (e.g., B) and, if necessary, invites them to join its local PKI. Suppliers (e.g. B) that themselves have direct suppliers (e.g. D and E are suppliers to B), initiate a recursive LCA for the product they deliver to their consumers (e.g. A). After B obtains the results of its group LCA, it provides this as input "up-stream" to the LCA initiated by A.

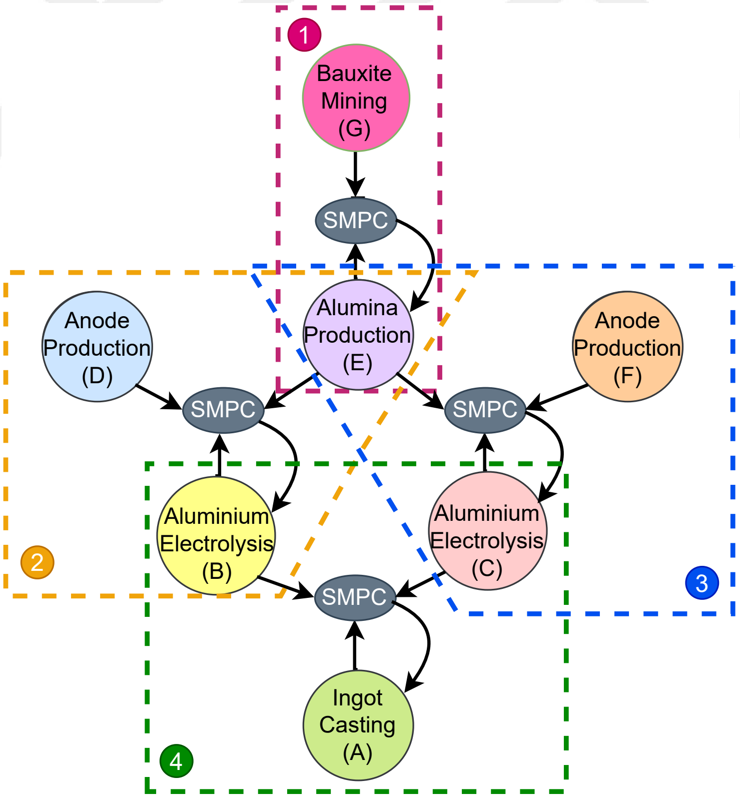


Figure 6.5: Recursive LCA Computation for Aluminium Production Supply Chain

Our implementation processes flows (collects data) from "cradle-to-grave" (up-stream collection) as shown in Figure 6.5. We first run an LCA computation for group 1 (companies E and G). In this case, we cannot secure shared data using SMPC, because company E has just one supplier (G). We ensure SMPC for the rest of the supply chain. Company E uses the result in up-stream computations as a supplier. We can execute the next two groups of LCA computations (group 2 and group 3) in parallel as they are independent of each other. The computation for group 2 is between the companies B, D, and E (using the result from the previous computation of group 1), and the computation for group 3 is between the companies C, F, and E. After B and C get the results from their group computations, they use these results for the final computation. We reach the resulting group 4 with the final consumer company A; the final computation happens between the companies A, B, and C, and the main consumer company A receives the LCA of the aluminium production supply chain.

6.4.2 Calculation of Enhanced LCA

In our system, each company (consumer companies and supplier companies) that joins a computation group supplies one unit of production of its environmental flows to SMPC. As companies provide one unit value of their production, the consumer company in the computation provides a vector to SMPC. This vector includes the scaling factor of each company and it can be considered as vector s. As explained in the previous subsection, the computation of the scaling vector does not require the application of SMPC. This significantly eliminates the workload of both inverting the local matrix A and calculating the scaling vector s for the LCA computation of each group.

As our process flows from "cradle-to-grave", we first make a computation for the small rectangle shown with *number* 1 in Figure 6.5. We cannot provide secure multi-party computation for the computation between company E and company G because company E (*Alumina production*) has just one supplier company (G). The following computation shows the aggregation between the companies of *Alumina Production* and *Bauxite Mining*. The columns of matrix $\mathcal{B}_1$ represent *Alumina Production* process (company E) and *Bauxite Mining* process (company G), respectively, and the rows of matrix $\mathcal{B}_1$ represent the number of environmental flows (Particulates, NO_2, SO_2, and Mercury (Hg), respectively) of each company. The s_1 scaling vector is provided by company E (as it is the consumer company for this group computation):

$$\mathcal{B}_1 = \begin{array}{c} \\ P \\ NO_2 \\ SO_2 \\ Hg \end{array} \!\! \begin{array}{c} \begin{array}{cc} \quad E & \quad\quad G \end{array} \\ \begin{pmatrix} 5 \cdot 10^{-4} & 1 \cdot 10^{-4} \\ 6 \cdot 10^{-4} & 0 \cdot 10^{0} \\ 11 \cdot 10^{-4} & 0 \cdot 10^{0} \\ 12 \cdot 10^{-5} & 0 \cdot 10^{0} \end{pmatrix} \end{array} \quad s_1 = \begin{pmatrix} 1 \cdot 10^{0} \\ 2\,847 \cdot 10^{-3} \end{pmatrix} \quad g_1 = \mathcal{B}_1.s_1 = \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} \begin{pmatrix} 78 \cdot 10^{-5} \\ 6 \cdot 10^{-4} \\ 11 \cdot 10^{-4} \\ 12 \cdot 10^{-5} \end{pmatrix} \tag{6.13}$$

However, we ensure secure multi-party computation for the rest of the supply chain. After company E gets the aggregated environmental flows (g_1), company E uses this in upper group computations as a supplier. The next group computations (group 2 and group 3) can be done in parallel because they are independent of each other. The computation in group 2 happens between companies B, D, and E (this will use the aggregation result g_1 in matrix $\mathcal{B}_2$), and the scaling vector s_2 is provided by company B:

$$\mathcal{B}_2 = \begin{array}{cc} & \begin{array}{ccc} B & D & E \end{array} \\ \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} & \begin{pmatrix} 1 \cdot 10^{-3} & 2 \cdot 10^{-4} & 78 \cdot 10^{-5} \\ 3 \cdot 10^{-4} & 5 \cdot 10^{-4} & 6 \cdot 10^{-4} \\ 132 \cdot 10^{-4} & 43 \cdot 10^{-4} & 11 \cdot 10^{-4} \\ 0 \cdot 10^{0} & 0 \cdot 10^{0} & 12 \cdot 10^{-5} \end{pmatrix} \end{array} \quad s_2 = \begin{pmatrix} 1 \cdot 10^{0} \\ 46 \cdot 10^{-2} \\ 1\,919 \cdot 10^{-3} \end{pmatrix} \quad g_2 = \mathcal{B}_1.s_1 = \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} \begin{pmatrix} 26 \cdot 10^{-4} \\ 168 \cdot 10^{-5} \\ 173 \cdot 10^{-4} \\ 23 \cdot 10^{-5} \end{pmatrix} \tag{6.14}$$

The computation in group 3 happens between companies C, F, and E (using the aggregation result g_1 in matrix $\mathcal{B}_3$), and the scaling vector s_3 is provided by company C:

$$\mathcal{B}_3 = \begin{array}{cc} & \begin{array}{ccc} C & F & E \end{array} \\ \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} & \begin{pmatrix} 2 \cdot 10^{-3} & 0 \cdot 10^{0} & 78 \cdot 10^{-5} \\ 4 \cdot 10^{-4} & 1 \cdot 10^{-5} & 6 \cdot 10^{-4} \\ 8 \cdot 10^{-3} & 1 \cdot 10^{-4} & 11 \cdot 10^{-4} \\ 0 \cdot 10^{0} & 0 \cdot 10^{0} & 12 \cdot 10^{-5} \end{pmatrix} \end{array} \quad s_3 = \begin{pmatrix} 1 \cdot 10^{0} \\ 515 \cdot 10^{-3} \\ 2\,105 \cdot 10^{-3} \end{pmatrix} \quad g_3 = \mathcal{B}_3.s_3 = \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} \begin{pmatrix} 36 \cdot 10^{-4} \\ 17 \cdot 10^{-4} \\ 10 \cdot 10^{-3} \\ 25 \cdot 10^{-5} \end{pmatrix} \tag{6.15}$$

The computation in group 4 is the last computation and involves the main consumer company A and its direct supplier companies B and C. Suppliers B and C use the aggregation results (g_2 and g_3) obtained from the previous computations of group 2 and group 3. The final scaling vector (s_4) is provided by the main consumer company A:

$$\mathcal{B}_4 = \begin{array}{cc} & \begin{array}{ccc} A & B & C \end{array} \\ \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} & \begin{pmatrix} 4 \cdot 10^{-5} & 26 \cdot 10^{-4} & 36 \cdot 10^{-4} \\ 8 \cdot 10^{-5} & 168 \cdot 10^{-5} & 17 \cdot 10^{-4} \\ 3 \cdot 10^{-5} & 173 \cdot 10^{-4} & 10 \cdot 10^{-3} \\ 0 \cdot 10^{0} & 23 \cdot 10^{-5} & 25 \cdot 10^{-5} \end{pmatrix} \end{array} \quad s_4 = \begin{pmatrix} 1 \cdot 10^{0} \\ 95 \cdot 10^{-2} \\ 5 \cdot 10^{-2} \end{pmatrix} \quad g_4 = \mathcal{B}_3.s_3 = \begin{array}{c} P \\ NO_2 \\ SO_2 \\ Hg \end{array} \begin{pmatrix} 269 \cdot 10^{-5} \\ 176 \cdot 10^{-5} \\ 1\,697 \cdot 10^{-5} \\ 23 \cdot 10^{-5} \end{pmatrix} \tag{6.16}$$

The inventory vector $g4$ gives the result for 1 kg of *Primary Aluminium*. The system emits 2.69 kg of particulates, 1.76 kg of NO_2, 16.9 kg of SO_2, and 0.23 g of mercury (Hg) in producing 1000 kg of *Primary Aluminium*.

6.5 Security and Privacy Analysis of Enhanced LCA Model

After designing an inter-organizational system for a confidentiality-enhanced LCA that is based on our threat modelling approach, in the following, we move on to discussing the mitigation strategies that our model provides and the accepted threats.

6.5.1 Mitigating Threats

To ensure the confidentiality of the economic and environmental flows in an LCA computation, we use SMPC. SMPC helps us to achieve the confidentiality of data entered into the LCA computation and, in addition, avoids the need for a trusted third-party. The drawback is that SMPC is slow; applying it naïvely to the whole collaboration results in a system that is too slow to be used in practical applications. Hence, we designed the system in such a way that each direct supplier–consumer group performs a secure LCA, and the consumer pushes this result upstream. We consider this an acceptable solution, as there is a certain trust relationship between a consumer and its direct suppliers. By breaking the computation up into local sub-computations, we not only bring the performance down to near real-time (benchmark results are provided in Chapter 7), but we also remove the need to reveal sub-suppliers to a consumer company. By this, we mitigate the threat of breaking the confidentiality of the consumer–supplier relationship. In addition, the communication within each group is secured by a PKI (operated by the local consumer company) that companies can join using pseudonyms, providing anonymity between different suppliers to the same consumer company.

In the following, we go into more detail about addressing the threats as laid out in Subsection 6.3.2.

Addressing Attacks from Downstream Threat Actors

A consumer company as an attacker could want to know the suppliers (membership), their business relationships, and the confidential data of a specific supplier (their production processes and environmental flows) in the supply chain. We mitigate the following attacks from Section 6.3.2:

- *Competitive/Complementor Sub-suppliers Membership*: In our approach, we propose a hierarchical grouping method that reduces the number of available communication channels between supply chain participants. By limiting communication to only those participants with a direct relationship, our approach prevents consumers from gaining knowledge about indirect suppliers (downstream). Additionally, we establish a public-key infrastructure for each group, with the identities of companies authenticated by the consumer company responsible for the group, ensuring that consumers cannot access the membership information of companies outside their group.
- *Business Relationships of Suppliers*: The hierarchical grouping in our approach organizes the supply chain into distinct groups, each comprising a consumer company and its direct suppliers. In this way, suppliers provide information relevant to the specific group they are a part of, ensuring that consumer companies can only access information about business relationships that they are directly a part of, and not those in which they have no involvement.

- *Production Processes of Suppliers*: In order to prevent consumers from learning the production processes of suppliers in a group and mitigate attacks on them, we use the application of SMPC. SMPC gives us the guarantee that companies can only know their own input in the computation they are involved in and potentially the final outcome of the computation.
- *Environmental Flows of Suppliers*: To mitigate attacks on the data confidentiality of suppliers or internal production details, we mainly rely on the application of SMPC. In each computation group, the consumer company of a specific group decides the list of environmental flows collected from its direct suppliers. The amounts of these environmental flows are aggregated via SMPC, and then the consumer company gets the total amount from SMPC. In SMPC, the inputs of each company remain private and are not revealed to any other company during the computation. Therefore, the consumer company cannot access the individual values of the environmental flows of suppliers.

Addressing Attacks from Upstream Threat Actors

An insider attacker in a supplier role may be interested in obtaining information about consumer companies, including identifying the companies themselves, understanding their relationships with other businesses, determining their production levels, and discovering the materials used in the manufacture of their products. In the following, we describe how we mitigate these attacks (given in Section 6.3.2) in our system:

- *Consumer Membership*: Our hierarchical grouping approach avoids indirect communications between companies in the supply chain. As a result, a supplier company is aware of its direct consumer; however, it does not have access to information about other consumer companies located in different hierarchical groups within the supply chain.
- *Business Relationships of Consumer*: Our approach sets up a PKI for each group. This PKI issues certificates to each participant in the group, which are assigned by the group's consumer. In order to securely communicate during the computation, these certificates are shared among all participants in the group. However, participants use pseudonyms to protect their identity and anonymity. This prevents disclosing the business relationships of consumer in the group and ensures that the business relationships of the group's consumer remain confidential.
- *Production Process of Consumer*: In our approach, we require that companies joining a group computation enter one unit value of their production to the computation; the consumer company of that computation group locally generates the scale vector of the computation group, representing scaling factors for companies in the computation, and does not share the vector with the suppliers. When the LCA computation happens for the group, this scale vector

is given to SMPC as secret information and kept secret during the computation as SMPC guarantees. Therefore, suppliers are not able to know the production processes (economic flows) of their consumers.

- *Environmental Flows of Consumer*: To mitigate attacks on shared data confidentiality, mainly for suppliers but also for consumers, we rely on the application of SMPC. The guarantee of SMPC is that, as long as at least one company follows the protocol correctly and no more than a threshold number of companies are corrupted, the output of the computation will be correct, and private inputs of the companies remain private. Additionally, the use of pseudonymous certificates in the LCA computation makes it impossible for suppliers to identify one another, greatly reducing the likelihood of collusion.

Addressing Attacks from Lateral Threat Actors

A supplier acting as an attacker may attempt to gain access to confidential information of other suppliers, such as their identities, the products they supply, the quantities they supply (and to whom), and the emissions generated by their products (environmental flows) (Section 6.3.2):

- *Competitor/Complementor/Sub-supplier of Other Suppliers Membership*: In our hierarchical grouping, sub-suppliers of suppliers are not included in a group if they do not have a direct relationship with the consumer. Additionally, the PKI establishment, allowing companies to use pseudonymous handles for their certificates, minimizes the risk of suppliers learning about each other. A company could further obfuscate their participation by adopting the roles of multiple suppliers (this requires cooperation by the consumer company acting as CA). Similarly, the consumer company can minimize the risk of revealing how many suppliers it has by adding artificial suppliers using pseudonyms.
- *Business Relationships of Other Suppliers*: In a hierarchical group within the supply chain, suppliers might be able to know that all participating companies in the group have a relationship with the group's consumer. However, a supplier cannot know the other business relationships of those companies may have within the supply chain.
- *Production Processes of Other Suppliers*: In our approach, suppliers are unable to access information about the production quantities of other suppliers. This is because suppliers only share data related to a single unit of their production using SMPC, and this information is kept confidential within SMPC. The consumer company, as part of a computation group, generates a scale vector that includes the scale factor for each of the materials it requires. This vector is also treated as confidential information and is kept secret during computation. As a result, our approach ensures that all production processes are protected.

- *Environmental Flows of Other Suppliers*: As previously stated, we use SMPC to perform secure calculations on shared data. Companies in a group, including the consumer company, collaborate on these computations, and the consumer company receives the computation results from SMPC. Therefore, suppliers do not have access to information about the environmental flows of other suppliers during the computation.

Mitigating Other Types of Threats

Although our approach focuses more on the confidentiality of the shared data during the LCA computation of a supply chain, it also helps to reduce or mitigate some cases of the other types of identified threats that supply chain systems can have. The detailed explanation of this is provided in Section 4.3. Also, in Section 3.4, we outlined potential mitigation strategies for threats that our approach does not address.

6.5.2 Accepting Threats

In our current implementation, we do not have mitigation for companies providing incorrect input into the LCA – i.e. we can not prevent tampering with input data. We accepted that risk, as the consumer company is likely to be able to detect data tampering and can utilize the contract between them and their suppliers to follow up legally. In a future version, we plan to investigate whether this threat can also be mitigated at a technical level, e.g. by using cryptographic commitment schemes.

In our model, we ensure that companies are not able to learn or disclose any information from an LCA computation since SMPC guarantees that the participants of a computation only know their own inputs and the result of the computation if the result is available to them. However, we cannot prevent the disclosure of the information that participants of a supply chain can infer from the information learned within one or several LCAs. It is hard to mitigate this type of threat at a technical level. Therefore, in the future, we plan to extend our threat analysis to include such inferred information, further supporting companies in their decision to join (or not join) a supply chain.

6.6 Summary

In this chapter, we presented a security and privacy analysis of the LCA of supply chains and presented a novel scalable privacy-preserving model for the LCA of supply chains that preserves the confidential information of participants while minimizing performance impact. Our threat modelling analysis of LCA revealed that the data sharing required for traditional LCA can expose various severe security and privacy risks within the supply chain. With our scalable privacy-preserving model, we improve the confidentiality of data of participants required during the computation of the LCA of supply chains. The approach is based on a combination of decomposing LCA computation and the use of SMPC. The model we presented for the LCA of supply chains minimizes the data being shared between participants of the supply chain, thereby enhancing the confidentiality of the data necessary for LCA. We anticipate that this will also improve performance. To determine the extent of this improvement, we will conduct a thorough evaluation in the next chapter.

7. Performance Evaluation

In this chapter, we will conduct a comprehensive performance evaluation of our approach using the LCA scenario introduced in the previous chapter. Firstly, we will present details regarding the experimental setup implemented for this evaluation. Following that, we will divide the rest of this chapter into two parts as follows.

Before evaluating the performance of our approach, we aim to determine the operational cost of conducting an LCA computation and evaluate the impact of applying SMPC directly to such computations. Therefore, in the following sections, from Section 7.2 to Section 7.4, we will initially focus on examining the traditional LCA method with the direct application of SMPC. We will begin by outlining the time complexity of individual operations within a traditional LCA computation, as well as the overall time complexity of a traditional LCA computation as a whole in Section 7.2. Subsequently, we will identify the parts or operations of a traditional LCA computation that are performed using SMPC to preserve the security and confidentiality of data necessary for completing an LCA (in Section 7.3). Lastly, in Section 7.4, we will evaluate the performance of directly applying SMPC to LCA computations.

After examining the performance of the traditional LCA using SMPC naively, we will present the performance of our own approach, a confidentiality-enhanced LCA model. Our aim is to determine and demonstrate the cost of running LCA scenarios with our enhanced LCA model. To achieve this, we will follow a similar structure in Section 7.5 through Section 7.7 as before for the traditional LCA. First, we will outline the time complexity of operations within our enhanced LCA model, as well as the overall time complexity of the enhanced LCA computation in Section 7.5. Next, in Section 7.6, we will show which parts or specific operations in our enhanced LCA computation are performed using SMPC to maintain the security and confidentiality of data required to complete the enhanced LCA. We will demonstrate how our enhanced LCA model, which utilizes a hierarchical grouping approach, applies SMPC to LCA computations and enhances the performance of those computations. Finally, we will present the runtime results of our model through benchmarking in Section 7.7. We will demonstrate how our approach improves the confidentiality of LCAs without sacrificing performance.

It is important to note that while some parts of the performance evaluation are specific to LCA, the majority of it can be easily applied to other scenarios. Thus, the performance evaluation presented in this chapter is not limited to LCA scenarios. We will provide further details in the following sections.

7.1 Experimental Setup

All the experiments presented in this chapter are based on the implementation we described earlier in Chapter 5. We created a prototype using SCALE-MAMBA [140] as the SMPC implementation. The remaining infrastructure, such as the creation of PKIs, the distribution of SMPC programs to suppliers, and the execution of local computation by each participant, was implemented in Python.

We ran the experiments on a server with a Xeon E4-2680v4 CPU (with 28 CPU threads, i.e. 14 physical cores) and 256 GB of RAM. All experiments were designed in such a way that concurrently running processes on that one machine (the server) have enough resources.

During the evaluation, our goal is to demonstrate how our approach improves the performance of computation within business collaborations compared to the direct application of SMPC, specifically focusing on the impact of hierarchical grouping. In our experiments, we idealised network communication by using the local network interface. With the local network setup, we simulate a network with low latency and high bandwidth. Furthermore, with this setup, we can guarantee that network behaviour is unlikely to change or be disrupted by other processes between different experiments and even during a single experiment.

7.2 Traditional LCA: Time Complexity

In Subsection 6.1.1, we presented a case study to demonstrate the calculation steps involved in performing a traditional LCA. In the following, we present the time complexity of these steps, as well as the time complexity of the traditional LCA as a whole, to determine the cost of operations in a traditional LCA computation. We provide the theoretical runtime analysis of the algorithms that we execute on top of the framework, SCALE-MAMBA, because we would like to understand what the theoretical lower bound (without using SMPC) is that we can expect to observe in our experiments. This lower bound helps us to understand the overall runtime behaviour in relation to the runtime behaviour without SMPC.

Calculation of Inverse of Matrix $\mathcal{A}$

In a traditional LCA, the first step is to calculate the inverse of matrix $\mathcal{A}$ (Subsection 6.1.1). The main consumer company, which is responsible for this computation, creates a matrix, matrix $\mathcal{A}$, that represents the economic flows of the participants in the LCA supply chain. This matrix is then inverted to obtain matrix $\mathcal{A}^{-1}$, which is used in the calculation of the scaling vector s. In Algorithm 2, we provide a pseudocode representation of the inverse of matrix $\mathcal{A}$ using the Gauss Jordan method [157], which is relatively straightforward to understand and implement. This pseudocode presents a detailed breakdown of the process for inverting the matrix and helps to understand the runtime complexity of the computation. In our experiments, the actual goal is to demonstrate how our approach enhances computation performance compared to the direct application of SMPC in various supply chain scenarios with varying supply chain structures. We aim to show that the idea of grouping in supply chain scenarios, including hierarchical groups, improves performance. To facilitate a fair comparison, we use the same matrix inversion algorithm– the Gauss-Jordan method–for both our approach and the direct SMPC application. Consequently, the complexity of the chosen method for matrix inversion does not affect our goal. In the following, we will determine the time complexity of the operations involved in computing the inverse of matrix $\mathcal{A}$. The *numbers underlined in the equations* represent the *line numbers* where the corresponding code begins.

To compute the inverse of matrix $\mathcal{A}$, first of all, the companies responsible for each unit process in the supply chain provide their inputs, which are economic flows, to matrix $\mathcal{A}$. The number of unit processes is equal to the number of economic flows, which is n (as detailed in Subsection 6.1.1). This step involves entering inputs into a 2D array, which requires two nested `for` loops. This is represented in lines of code from 2 to 6. The statement in line 4 is a part of the second `for` loop, which has the same time complexity as the second loop. As each loop is run n times, the time complexity of this operation is:

$$\begin{aligned} T(n) &= t(forloop_2) \times [t(forloop_3) + t(statement_4)] \\ &= n \times [n + n] = n \times [2n] = 2n^2 \Rightarrow O(n^2) \end{aligned} \tag{7.1}$$

Then, we augment the identity matrix to matrix $\mathcal{A}$. In this operation, we have two nested loops and one `if` statement placed in the second `for` loop. The process is demonstrated in the lines of code from 8 to 16. In this case, it also results in a time complexity of $O(n^2)$:

$$\begin{aligned} T(n) &= t(forloop_8) \times [t(forloop_9) + t(ifstatement_{10})] \\ &= n \times [n + n] = n \times [2n] = 2n^2 \Rightarrow O(n^2) \end{aligned} \tag{7.2}$$

Algorithm 2 Pseudocode for Inverse of Matrix $\mathcal{A}$ Using Gauss Jordan Method

```
▷ Read Matrix $\mathcal{A}$
for $i \leftarrow 1, n$ do
    for $j \leftarrow 1, n$ do
        Read $\mathcal{A}[i, j]$
    end for
end for
▷ Augment Identity Matrix to Matrix $\mathcal{A}$
for $i \leftarrow 1, n$ do
    for $j \leftarrow 1, n$ do
        if $i = j$ then
            $\mathcal{A}[i, j + n] = 1$
        else
            $\mathcal{A}[i, j + n] = 0$
        end if
    end for
end for
▷ Apply Gauss–Jordan Elimination on Augmented Matrix $\mathcal{A}$
for $i \leftarrow 1, n$ do
    if $\mathcal{A}[i, i] = 0$ then
        $Mathematical\ Error$
        $Stop$
    end if
    for $j \leftarrow 1, n$ do
        if $i \neq j$ then
            $Ratio = \mathcal{A}[j, i]/\mathcal{A}[i, i]$
            for $k \leftarrow 1, 2 * n$ do
                $\mathcal{A}[j, k] = \mathcal{A}[j, k] - Ratio * \mathcal{A}[i, k]$
            end for
        end if
    end for
end for
▷ Normalize Principal Diagonal
for $i \leftarrow 1, n$ do
    for $j \leftarrow n + 1, 2 * n$ do
        $\mathcal{A}[i, j] = \mathcal{A}[i, j]/\mathcal{A}[i, i]$
    end for
end for
▷ Create Inverse Matrix of $\mathcal{A}$
for $i \leftarrow 1, n$ do
    for $j \leftarrow n + 1, 2 * n$ do
        $\mathcal{A}^{-1}[i, j] = \mathcal{A}[i, j]$
    end for
end for
```

To calculate the inverse of matrix $\mathcal{A}$, we use the Gauss–Jordan elimination method on an augmented version of matrix $\mathcal{A}$. This process involves three nested loops, which contribute to a time complexity of $O(n^3)$. This means that the time required to execute this operation increases at a cubic rate in proportion to the size of the input. In more detail:

$$\begin{aligned} T(n) &= t(forloop_{18}) \times [t(forloop_{23}) \times [t(forloop_{26}) + t(statement_{27})] \\ &\quad + t(ifstatement_{24})] + t(ifstatement_{19}) \\ &= n \times [n \times [2n + 2n] + n] + n = n \times [4n^2 + n] + n \\ &= 4n^3 + n^2 + n \Rightarrow O(n^3) \end{aligned} \tag{7.3}$$

Then, in lines 32 to 43 of code, we normalize the principal diagonal to obtain the inverse matrix, $\mathcal{A}^{-1}$. These two operations have the same time complexity as shown in Equation 7.1, which is $2n^2$. As a result, the calculation of the inverse of matrix $\mathcal{A}$ has a time complexity of $O(n^3)$, as demonstrated in Equation 7.4 (which is also stated in [157]):

$$\begin{aligned} T(n) &= 2n^2 + 2n^2 + 4n^3 + n^2 + n + 2n^2 + 2n^2 \\ &= 4n^3 + 9n^2 + n \Rightarrow O(n^3) \end{aligned} \tag{7.4}$$

Calculation of Vector s

In a traditional LCA, after obtaining the inverse of matrix $\mathcal{A}$, we need to calculate the scaling vector s to determine the scaling factor for each unit process in the supply chain. To do this, we perform the multiplication of the inverse of matrix $\mathcal{A}$, $\mathcal{A}^{-1}$, and the final demand vector, f. In Algorithm 3, we present a pseudocode representation of the operations involved in computing scaling vector s to help us determine the runtime complexity of the calculation of scaling vector s. Final demand vector f includes reference flows, representing the quantity of material(s) produced by the supply chain. Even if only one company is the reference flow, this vector includes the complete set of economic flows, which is n.

Since we have already obtained the inverse of matrix $\mathcal{A}$, we exclude the runtime step of reading the inverse matrix $\mathcal{A}^{-1}$, which is lines 1 to 6. In a traditional LCA, the final demand vector is given to the system by the main consumer company in the supply chain. This step has a time complexity of $O(n)$:

$$T(n) = t(forloop_8) + t(statement_9)] = n + n = 2n \Rightarrow O(n) \tag{7.5}$$

Algorithm 3 Pseudocode for Calculation of Scaling Vector s

```
▷ Read the Inverse Matrix $\mathcal{A}^{-1}$
for $i \leftarrow 1, n$ do
    for $j \leftarrow 1, n$ do
        Read $\mathcal{A}^{-1}[i,j]$
    end for
end for
▷ Read Final Demand Vector $f$
for $i \leftarrow 1, n$ do
    Read $f[i,0]$
end for
▷ Calculate Scaling Vector $s$
for $i \leftarrow 1, n$ do
    for $j \leftarrow 1, n$ do
        $s[i,0] = s[i,0] + \mathcal{A}^{-1}[i,j] * f[j,0]$
    end for
end for
```

Scaling vector s is then computed in the lines of code between 11 and 16. This step includes two nested `for` loops and one statement that depends on the size of the loops. As a result, the time complexity of this process's calculation is $O(n^2)$:

$$\begin{aligned} T(n) &= t(forloop_{12}) \times [t(forloop_{13}) + t(statement_{14})] \\ &= n \times [n + n] = 2n^2 \Rightarrow O(n^2) \end{aligned} \tag{7.6}$$

As shown in Algorithm 3, the time complexity of the calculation of scaling vector s is $O(n^2)$:

$$T(n) = 2n + 2n^2 \Rightarrow O(n^2) \tag{7.7}$$

Calculation of Vector g

As the final step in a traditional LCA computation, we compute the inventory vector g. This vector comprises the system-wide aggregated environmental flows and is derived through the multiplication of matrix $\mathcal{B}$ by scaling vector s. Before performing this computation, participants enter their environmental flows into matrix $\mathcal{B}$. This multiplication yields the final inventory vector, which is an important output of the LCA process.

The time complexity of this multiplication is determined by the sizes of matrix $\mathcal{B}$ and scaling vector s, which are represented by the values of m and n, respectively. m represents the number of environmental flows involved in the supply chain, while n represents the number of unit processes. The pseudocode representation for inventory vector g is provided in Algorithm 4.

Algorithm 4 Pseudocode for Calculation of Inventory Vector g

```
 1: ▷ Read Matrix $\mathcal{B}$
 2: for $i \leftarrow 1, m$ do
 3:     for $j \leftarrow 1, n$ do
 4:         Read $\mathcal{B}[i, j]$
 5:     end for
 6: end for
 7: ▷ Read Scaling Vector $s$
 8: for $i \leftarrow 1, n$ do
 9:     Read $s[i, 0]$
10: end for
11: ▷ Calculate Inventory Vector $g$
12: for $i \leftarrow 1, m$ do
13:     for $j \leftarrow 1, n$ do
14:         $g[i, 0] = g[i, 0] + \mathcal{B}[i, j] * s[j, 0]$
15:     end for
16: end for
17: ▷ Display Inventory Vector $g$
18: for $i \leftarrow 1, m$ do
19:     $Print\ g[i, 0]$
20: end for
```

The lines of code from 1 to 6 represent entering inputs for matrix $\mathcal{B}$, and the time complexity of this step is:

$$\begin{aligned} T(n) &= t(forloop_2) \times [t(forloop_3) + t(statement_4)] \\ &= m \times [n + n] = 2mn \Rightarrow O(mn) \end{aligned} \tag{7.8}$$

Since we already have scale vector s, we excluded the time complexity of reading this vector. Then, we get to our main aim, which is to compute the inventory vector, g. As seen from the lines of code from 12 to 16, the step of computing the inventory vector includes two nested for loops and one statement, which contribute to a time complexity of $O(mn)$:

$$\begin{aligned} T(n) &= t(forloop_{12}) \times [t(forloop_{13}) + t(statement_{14})] \\ &= m \times [n + n] = 2mn \Rightarrow O(mn) \end{aligned} \tag{7.9}$$

Finally, printing the result requires $O(m)$ time complexity; the lines of code from 18 to 20 are:

$$T(n) = t(forloop_{18}) + t(statement_{19})] = m + m = 2m \Rightarrow O(m) \tag{7.10}$$

As a result, the time complexity of computing of inventory vector g is $O(mn)$ (Equation 7.12):

$$T(n) = 2mn + 2mn + 2m = 4mn + 2m \Rightarrow O(mn) \tag{7.11}$$

Putting Everything Together

In Table 7.1, we illustrate the time complexities of the steps involved in calculating a traditional (standard) LCA. As shown in the table, the most costly operation in LCA computation is the *inversion of matrix $\mathcal{A}$*. In this table, n is the number of production/units processes in the supply chain, which is also equal to the number of economic flows, and m is the number of environmental flows in the supply chain.

Table 7.1: Time Complexity of Operations in Traditional LCA

Operation	**Input**	**Output**	**Complexity**
Calculation of inverse of matrix $\mathcal{A}$	One $n \times n$ matrix ($\mathcal{A}$)	One $n \times n$ matrix ($\mathcal{A}^{-1}$)	$O(n^3)$
Calculation of scaling vector s	One $n \times n$ matrix ($\mathcal{A}^{-1}$) One $n \times 1$ matrix (f)	One $n \times 1$ matrix (s)	$O(n^2)$
Calculation of inventory vector g	One $m \times n$ matrix ($\mathcal{B}$) One $n \times 1$ matrix (s)	One $m \times 1$ matrix (g)	$O(mn)$

By combining the equations in Equation 7.4, Equation 7.7, and Equation 7.12, we can determine the total time complexity of traditional LCA to be $O(n^3)$:

$$\begin{aligned} T(n) &= [4n^3 + 9n^2 + n] + [2n + 2n^2] + [4mn + 2m] \\ &= 4n^3 + 11n^2 + 4mn + 3n + 2m \Rightarrow O(n^3) \end{aligned} \tag{7.12}$$

7.3 Traditional LCA: Applying SMPC

To ensure the security and confidentiality of data in traditional LCA, it is important to perform all calculation steps in SMPC. This is because the matrices ($\mathcal{A}$, $\mathcal{B}$, f, s) involved in the calculation steps contain sensitive information that could be exploited in attacks or lead to the disclosure of confidential data. Figure 7.1 illustrates the flowchart of the calculation steps involved in a traditional LCA, with the parts that should be computed using SMPC highlighted.

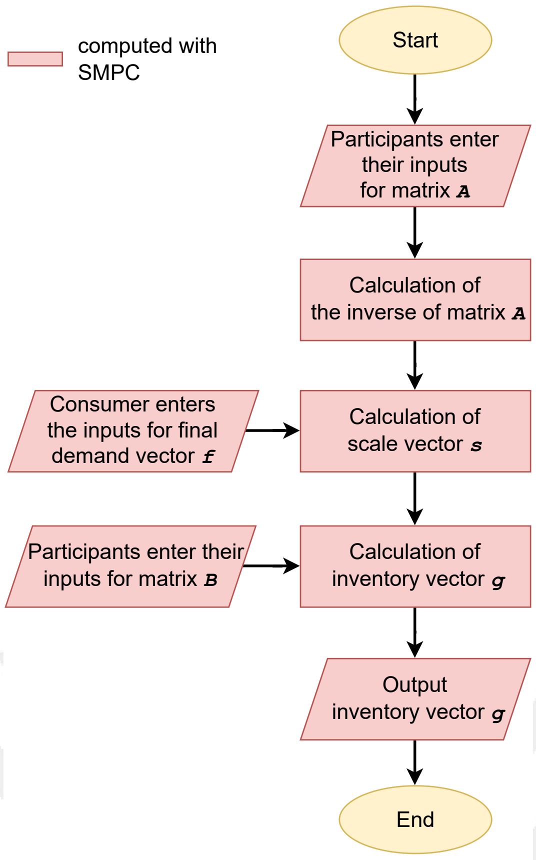

computed with SMPC
Start
Participants enter their inputs for matrix A
Calculation of the inverse of matrix A
Consumer enters the inputs for final demand vector f
Calculation of scale vector s
Participants enter their inputs for matrix B
Calculation of inventory vector g
Output inventory vector g
End

Figure 7.1: Flowchart of a Traditional LCA Using SMPC

7.4 Traditional LCA: Runtime Evaluation

While performing all these calculation steps with SMPC can protect against such threats, it can also significantly impact system performance. In this section, we aim to answer two important questions regarding the performance of LCA computation with direct application of SMPC: 1. What is the impact of applying SMPC directly to LCA computation? and 2. How does matrix inversion, as the most costly operation in LCA, impact the overhead of SMPC? To address these questions, we have designed two experiments as follows:

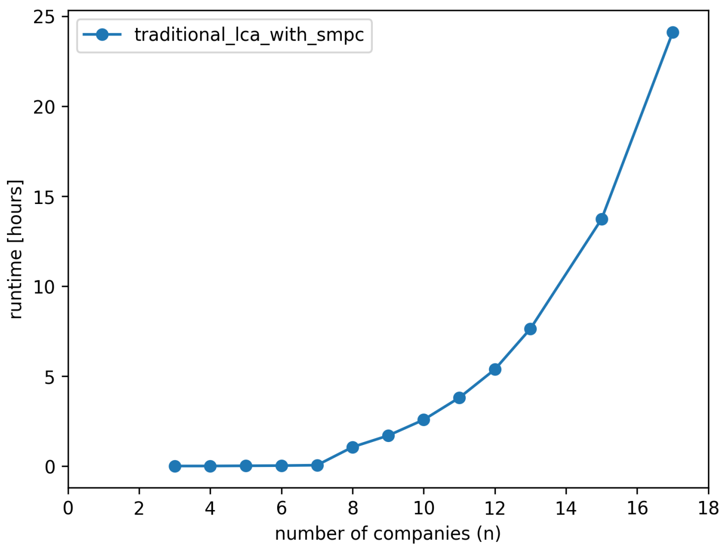
Experiment 1

One of the important questions we aim to address is the impact of applying SMPC directly to LCA computation. Our hypothesis is that the direct application of SMPC in LCA computation runs too slowly for practical use cases. Applying SMPC directly to LCA computation has a performance problem as it does not scale well with the number of companies due to added overhead. To answer this question and test this hypothesis, we have designed an experiment that performs LCA computation using SMPC directly for various numbers of companies. Table 7.2 presents the information regarding the experiment we performed, including the question, hypothesis, methodology, and outcome.

Table 7.2: Direct Application of SMPC on LCA

Experiment 1	
Question	What is the impact of applying SMPC directly to LCA computation?
Hypothesis	The direct application of SMPC in LCA computation runs too slowly for practical use cases.
Experiment	We designed an experiment that executes LCA computation using SMPC directly for various numbers of companies.
Result	SMPC adds a significant overhead to the runtime of LCA computation. As the number of participants in the LCA computation increases, the overhead of SMPC increases significantly.
Conclusion	The hypothesis is verified.

Figure 7.2 illustrates the performance of applying SMPC directly to the traditional LCA for various sizes of a supply chain. Our goal is to demonstrate the impact of SMPC on the runtime of the traditional LCA. The figure shows that as the number of companies in the supply chain (denoted by n) increases, the overall runtime still increases in a cubic way. This indicates that SMPC only adds a constant overhead in this particular scenario. However, this overhead is still significant.



n	$t[h]$
3	00:00:35
4	00:00:41
5	00:01:20
6	00:02:00
7	00:03:24
8	01:03:53
9	01:42:09
10	02:34:58
11	03:48:26
12	05:23:52
13	07:38:37
15	13:44:20
17	24:24:32

Figure 7.2: Runtimes of the Traditional LCA with SMPC

It is an important point to note that there is a significant increase in runtime when comparing the results between test scenarios involving 7 and 8 companies in the supply chain, as shown in the table in Figure 7.2. Specifically, the runtime of the

LCA scenario with 7 participants is 3 minutes and 24 seconds, while the runtime of the LCA scenario with 8 participants is 1 hour, 3 minutes, and 53 seconds. To assess the performance of the traditional LCA using SMPC, we used the prototype we developed for our approach. In our prototype, we use SCALE-MAMBA as the SMPC implementation. In the SMPC implementation, we use threshold secret sharing. We use Shamir secret sharing, and the threshold (t) is determined by the number of participating companies (n) in the computation, which is $n/2 - 1$ (as explained in Section 5.3). We chose this threshold based on the need for a guaranteed honest majority in order to ensure fairness and the delivery of the output through SMPC. In SCALE-MAMBA, the protocols change depending on t. If n is small, leading to a small t, then a non-interactive PRSS (Probabilistic Reliable Secret Sharing) is used to generate the random shares. On the other hand, when n – and thus t – is large, an interactive protocol is used instead, involving multiple rounds of communication between parties. Once there are eight or more participants, we reach a threshold that requires the use of an interactive protocol for generating random shares, which is the cause of the sudden increase in runtime results. However, it is important to note that this does not affect our findings. A detailed discussion of why our findings remain unaffected will be presented in Subsection 7.7.3.

Based on the experiment, we can conclude that the direct application of SMPC results in high overhead, not only for LCA scenarios but also in other scenarios involving multiple participants. As the number of participants in a scenario increases, so does the overhead of SMPC.

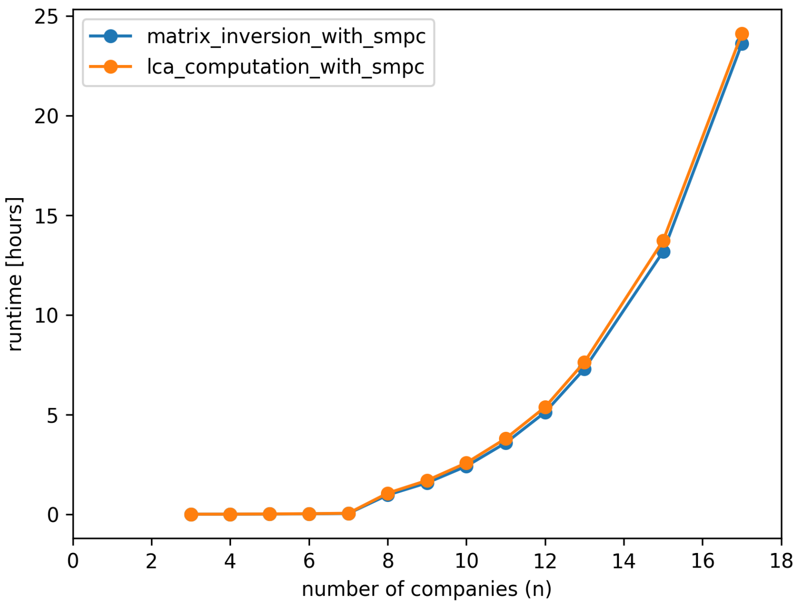
Experiment 2

We have seen that matrix inversion is the most costly operation in LCA computation. Furthermore, we are interested in determining the impact of matrix inversion on SMPC overhead. We hypothesize that with SMPC enabled, the vast majority of actual runtime is in the matrix inversion. We have designed an experiment to answer the question of how matrix inversion affects the overhead of SMPC. Our experiment shows the comparison of runtimes of matrix inversion to the overall runtimes of LCA computation using SMPC to provide insight into the impact of matrix inversion on SMPC overhead. The information regarding the experiment we performed is presented in Table 7.3.

Table 7.3: Impact of Matrix Inversion on the Overhead of SMPC

Experiment 2	
Question	How does matrix inversion affect the overhead of SMPC?
Hypothesis	With SMPC enabled, the vast majority of the runtime is in matrix inversion.
Experiment	We designed an experiment to get the runtime of matrix inversion and the overall runtime of LCA computation using SMPC for various numbers of companies.
Result	The majority of the computational time is spent on inverting matrices during LCA computation using SMPC. This means that matrix inversion is the primary source of SMPC overhead.
Conclusion	The hypothesis is verified.

Figure 7.3 illustrates the runtime results of matrix inversion in LCA computation using SMPC for various numbers of companies, as well as the overall runtime results. As can be observed from the data depicted in the diagram and the accompanying table, with SMPC, it is even clearer that matrix inversion is a significant portion of the overall runtime, which is highlighted by the value of $t_{portion}$ in the table. In scenarios with more than 8 companies, it constitutes 91 to 98 per cent of the total runtime. For scenarios with fewer than 8 companies, it accounts for 69 to 76 per cent of the total runtime. Therefore, if we can reduce the size of the matrix to be inverted, we can expect a significant decrease in runtime. Our proposed hierarchical grouping method effectively achieves this. With our hierarchical grouping, we decrease the size of the matrix to be inverted. By implementing this method, it is expected that a considerable reduction in the overall runtime can be achieved. In the following sections, the impact of this optimization will be analysed in detail.



n	$t_{inversion}[h]$	$t_{total}[h]$	$t_{portion}(\%)$
3	00:00:24	00:00:35	68
4	00:00:28	00:00:41	68
5	00:00:58	00:01:20	72
6	00:01:32	00:02:00	76
7	00:02:36	00:03:24	76
8	00:58:21	01:03:53	91
9	01:34:55	01:42:09	93
10	02:25:00	02:34:58	94
11	03:35:17	03:48:26	94
12	05:07:23	05:23:52	95
13	07:18:09	07:38:37	95
15	13:11:02	13:44:20	96
17	23:37:05	24:24:32	98

Figure 7.3: Comparison of Matrix Inversion and Traditional LCA Performance in SMPC

In addition, it can be inferred from this experiment that the most complex operation involved in collaboration tends to generate the greatest amount of overhead in SMPC. This finding holds true not only for the LCA scenario but also for other similar scenarios.

7.5 Confidentiality-Enhanced LCA: Time Complexity

While SMPC can be an effective method for ensuring the security of LCA computations, it can also significantly impact the performance of these computations, leading to increased computation time and computational overhead. To overcome this challenge, we applied our novel approach to LCA. As described in Subsection 6.4.1, we have developed a scalable privacy-preserving model for LCA computations that enhances the confidentiality of data required for completing LCA without causing significant performance overhead. Our model differs from the traditional LCA method by introducing a hierarchical grouping approach, where supply chain participants are separated into sub-groups based on their relationships and roles in the supply chain. We then apply SMPC to each of these sub-groups individually.

In this section, we present the time complexity of the operations in our confidentiality-enhanced LCA computation to determine the cost of these operations, as well as the cost of our enhanced LCA computation. Our aim is to demonstrate how our hierarchical grouping reduces the impact of costly operations in LCA computation. Our approach leads to a decrease in the number of participants involved in any particular SMPC group, and also simplifies the calculations that should be computed using SMPC. Our approach allows us to reduce the time complexity of the joint LCA computation given in Equation 7.12 within one group i to:

$$\begin{aligned} T(n_i) &= [4n_i^3 + 9n_i^2 + n_i] + [2n_i + 2n_i^2] + [4m_i n_i + 2m_i] \\ &= 4n_i^3 + 11n_i^2 + 4m_i n_i + 3n_i + 2m_i \Rightarrow O(n_i^3) \end{aligned} \tag{7.13}$$

The number of economic flows (participants) in group i is designated as n_i, and the number of environmental flows in group i is designated as m_i. It is important to note that the number of participants in a group, n_i, may differ between groups. As a result, depending on the number of groups in the supply chain, the number of participants in a group, n_i, may be less than or equal to the total number of participants in the supply chain, represented by n in the time complexity of the traditional LCA in Section 7.2. Also, m_i may differ from the value of m used to represent the time complexity of the traditional LCA. This is because a group may have a specific interest in knowing either more or fewer of the environmental flows within their own group, as compared to the entire supply chain as a whole. The total time complexity of our enhanced LCA computation is derived from the aggregation of the time complexities of the groups within the supply chain. The number of groups is denoted as I:

$$T(n_1, n_2, ..., n_I) = O(n_1^3) + O(n_2^3) + ... + O(n_I^3) \tag{7.14}$$

Additionally, if companies agree to provide data for one unit of their product, then we can compute the environmental impact for one unit of production. In this case, the final demand vector f will have no effect on the scaling vector, s. The scaling factors for companies in scaling vector s will have the same coefficient as the consumer's local economic flows. This eliminates the overhead associated with computing the inverse of matrix $\mathcal{A}$ and the scaling vector s for each group, which have time complexities of $O(n_i^3)$ and $O(n_i^2)$, respectively. For more information, please refer to Section 6.4, which delves into the detail of our approach. After the elimination, the new time complexity of each group reduces to $O(m_i n_i)$, which is the time complexity of computing inventory vector g for a group.

To demonstrate the time complexity of the operations in our approach, we have also included a significantly simplified version of our enhanced LCA model in the form of pseudocode in Algorithm 5. This pseudocode provides a high-level overview of the computation steps involved in our model, allowing for easy analysis of its time complexity. By eliminating the need to calculate the inverse of matrix $\mathcal{A}$ and the scaling vector s, the time complexity of our enhanced LCA computation is now determined solely by the number of groups (I) within the supply chain, their sizes (n_i), and the calculation of the inventory vector g of each group. Later on, we will analyse the impact of the size and number of groups, using different scenarios. There is also an overhead of implementing SMPC for computing the inventory vector g in each group. However, we excluded it at this stage. We will also discuss the overhead of implementing SMPC in Section 7.7.

Algorithm 5 Pseudocode for Our Enhanced LCA Computation Model

```
1:  ▷ If there are $I$ groups in a supply chain
2:  for $i \leftarrow 1, I$ do
3:      ▷ For local LCA computation of group $i$
4:      ▷ Request scaling vector $s$ as an input from the consumer of group $i$
5:      for $j \leftarrow 1, n_i$ do
6:          Read $s[j, 0]$
7:      end for
8:      ▷ Request their environmental flows as an input from each participant in
        group $i$, generating B matrix
9:      for $k \leftarrow 1, m_i$ do
10:         for $j \leftarrow 1, n_i$ do
11:             Read $B[k, j]$
12:         end for
13:     end for
14:     ▷ Calculate Inventory Vector $g$ of group $i$
15:     for $k \leftarrow 1, m_i$ do
16:         for $j \leftarrow 1, n_i$ do
17:             $g[k, 0] = g[k, 0] + B[k, j] * s[j, 0]$
18:         end for
19:     end for
20:     ▷ output the Inventory Vector $g$ in group $i$ to the consumer of group $i$
21:     for $i \leftarrow 1, m_i$ do
22:         *Print* $g[i, 0]$
23:     end for
24: end for
```

The number of groups included in the supply chain is designated as I. The number of participants in each group is designated as n_i, and the number of environmental flows is designated as m_i. Another factor that impacts the performance of our approach is the structure of the supply chain. The structure can determine whether certain group computations can be performed in parallel, thereby reducing the overall runtime of the supply chain. With this in mind, in Equation 7.15, we aim to assess the time complexity of a single group in our hierarchical model, as outlined in lines 5 to 23 of the code. The time complexity of one group is as follows:

$$\begin{aligned} T(n_i) &= t(forloop_5) + t(statement_6) \\ &+ t(forloop_9) \times [t(forloop_{10}) + t(statement_{11})] \\ &+ t(forloop_{15}) \times [t(forloop_{16}) + t(statement_{17})] \\ &+ t(forloop_{21}) + t(statement_{22}) \\ &= n_i + n_i + m_i \times [n_i + n_i] + m_i \times [n_i + n_i] + m_i + m_i \\ &= 2n_i + 4n_i m_i + 2m_i \Rightarrow O(n_i m_i) \end{aligned} \tag{7.15}$$

The structure of the LCA supply chain plays a crucial role in determining the feasibility of parallel computation of various groups. In cases where the supply chain structure is not suitable for performing computation groups in parallel, the overall time complexity of the enhanced LCA computation is obtained by aggregating the optimized time complexities of each computation group within the supply chain:

$$T(n_1, n_2, ..., n_I) = O(n_1 m_1) + O(n_2 m_2) + ... + O(n_I m_I) \tag{7.16}$$

7.6 Confidentiality Enhanced LCA: Applying SMPC

In this section, our aim is to demonstrate which parts or specific operations in our enhanced LCA computation are performed using SMPC to maintain the security and confidentiality of data. As we stated earlier, our approach involves using SMPC on each group individually. Therefore, the flowchart in Figure 7.4 shows the LCA computation process for one group in our enhanced LCA model, with the steps computed using SMPC highlighted.

When participants in a group agree to share data for one unit of production, the enhanced LCA approach, in contrast to the traditional LCA approach, does not require participants to directly enter their inputs into SMPC for performing all calculation steps. Instead, it involves the consumer company of the group generating matrix $\mathcal{A}$, calculating the inverse of matrix $\mathcal{A}$, and scaling vector s locally. This is because these calculations do not involve any confidential data from the group's participants. The consumer company then provides scaling vector s of the group to SMPC. Additionally, all companies in the computation group, including the consumer, provide their environmental flows data to SMPC to generate matrix $\mathcal{B}$. These inputs are kept confidential during computation. The group's inventory vector g is then calculated using SMPC, and the consumer receives the result, which is the aggregated environmental flows data of the group.

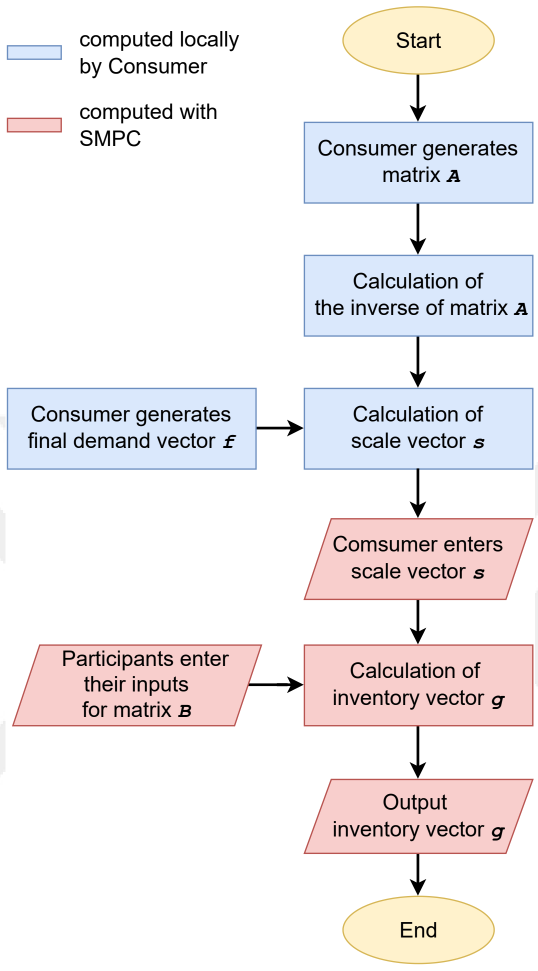

computed locally by Consumer
computed with SMPC
Start
Consumer generates matrix A
Calculation of the inverse of matrix A
Consumer generates final demand vector f
Calculation of scale vector s
Comsumer enters scale vector s
Participants enter their inputs for matrix B
Calculation of inventory vector g
Output inventory vector g
End

Figure 7.4: Flowchart of LCA Computation for One Group in the Enhanced LCA Model

7.7 Confidentiality Enhanced LCA: Runtime Evaluation

To evaluate the performance of our approach, we have designed a benchmarking study, which comprises various supply chain structures and methods for calculating LCA. The benchmarking scenarios have been carefully selected to provide a comprehensive performance evaluation of our approach. Before delving into the evaluation, we will provide an explanation for the selection of these scenarios in our benchmarking framework. Then, we will proceed to the experiments performed to evaluate the performance of our approach.

7.7.1 Selection of Benchmarking Scenarios

We evaluated our prototype using *three different supply chain scenarios* with varying *supply chain structures* (as shown in Figure 7.5).

- *Scenario 1* is a supply chain resembling a *balanced binary tree*, where each company has two direct suppliers (resulting in a local group size of three). Only companies on the last level have no suppliers. The total depth of the supply chain grows logarithmically with the total number of companies.
- *Scenario 2* simulates a supply chain resembling a linear list (a right-leaning tree), with each company having one supplier with no further suppliers and one company that again has two suppliers.
- *Scenario 3* resembles a "flat" supply chain with one consumer company and $n - 1$ suppliers.

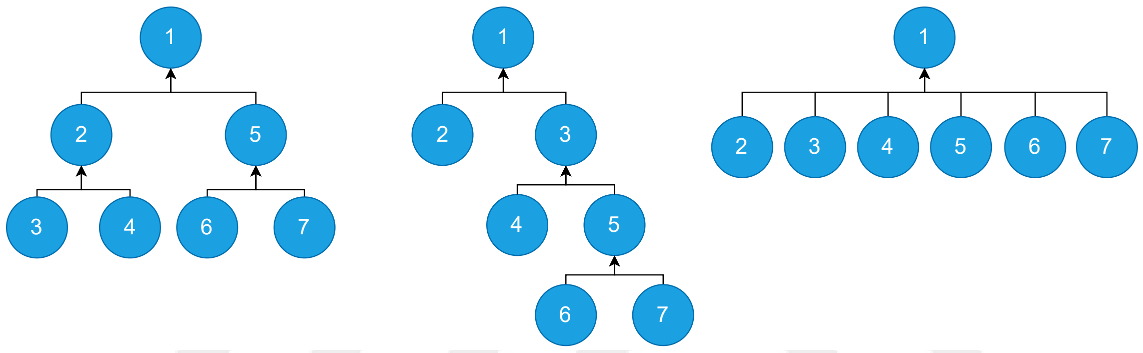


Figure 7.5: Structure of Supply Chain Scenarios 1–3 (From Left to Right)

We have chosen these three supply chain scenarios as they represent edge cases for our approach. The first scenario, a balanced binary tree, is the best case for our approach, as it allows for the maximum degree of parallelism for a given number of companies within a supply chain: all LCAs for groups on the same level can be executed in parallel. As it allows for parallel computations during the calculation of the LCA of a supply chain, we expect the runtime for the balanced binary tree scenario to grow roughly logarithmically with the number of companies. The second scenario, a right-leaning tree, represents the other extreme: all group computations need to be executed sequentially. In other words, computation groups need to wait for each other to execute their group computations. Therefore, we expect the runtime for the right-leaning tree scenario to grow roughly linearly, as one would expect for a supply chain whose length grows linearly. The third scenario, a flat tree, is used to study the impact of a different number of direct suppliers on the runtime of an individual SMPC-based LCA. We expect the runtime for the flat tree scenario to grow polynomially. As the third scenario models a "flat" supply chain, its runtime is mostly determined by one large SMPC over all inputs from all partners of the supply chain. Overall, we expect the structure of a real-world supply chain to be a mixture of scenarios 1 and 2. It is worth noting that in a real-world supply chain, not all business collaborations will have exactly two scenarios. In real-world supply chains, the number of participants in groups and the number of groups can vary. The number of participants in each group can be different. Also, there can be a supply chain where some groups can be executed in parallel, while others must be executed in sequence.

For each of these supply chain scenarios, we also used *three different methods for calculating LCA* to provide a more comprehensive evaluation:

- *Optimized LCA Calculation* is how our approach calculates the life-cycle assessment of a supply chain. This method allows dividing the supply chain into small groups, which consist of a consumer company and its direct suppliers, and then applying SMPC to the computations in each of these groups individually. This method also decreases the number of operations that need to be executed using SMPC within these computation groups. More information can be found in Section 7.5 and Subsection 6.4.2.
- *Full (Non-optimized) LCA Calculation* is a method that we included to create a variety of test scenarios for benchmarking. It involves assuming the existence of small groups, such as a consumer company and its direct suppliers. However, no calculation steps are performed locally by participating companies. Instead, all calculation steps in a group are computed using SMPC, including the determination of inverse of the $\mathcal{A}$ matrix, and scaling vector s and inventory vector g.
- *Traditional LCA Calculation* refers to the traditional method for performing LCA computation. It involves a single computation group, which incorporates inputs from all companies in the supply chain to determine the LCA for the entire supply chain. SMPC is implemented in this computation naively.

We have selected these three LCA calculation methods to demonstrate the enhanced performance that our approach offers. These methods allow us to assess the impact of the complexity of operations being computed and the number of participants being included on the runtimes of SMPC.

For an LCA scenario, we are aware of the operations need to be executed to obtain the results. We have identified and optimized the specific operations that must be performed in SMPC to obtain the desired results for the LCA of a supply chain. The first method, *optimized LCA calculation*, is used to show the performance of our enhanced LCA model. The second method, full LCA calculation, is used to examine the effect of the complexity of the operations being performed in SMPC. The third method *traditional LCA calculation*, serves to demonstrate the performance of a traditional LCA model that uses SMPC naively. This method also allows us to investigate the effect of the number of participants on the runtimes of SMPC.

The presented scenarios and methods focus on and encompass the fundamental aspects of our approach: hierarchical grouping, parallel computing, reduced number of participants, and decreased complexity or number of operations in SMPC. They assist in analysing the impact of implementing our approach on the performance of different collaboration scenarios. As such, they are between them sufficiently comprehensive to evaluate the performance of our approach.

7.7.2 Evaluation of the Performance of the Confidentiality-Enhanced LCA Model through Experiments

By considering a range of supply chain scenarios and using multiple calculation methods, we have carried out a comprehensive evaluation of the performance of our approach in the context of LCA. LCA is particularly relevant to our approach because its scenarios can incorporate a hierarchical structure and operations with different levels of complexity. We can vary the LCA scenarios to assess the performance of our approach comprehensively. In the following, we will present several experiments that enable us to evaluate and demonstrate the performance of our approach. Each experiment presents a range of different test results, which we analyse to evaluate the specific hypothesis we have proposed. To this end, we have run various test scenarios on our prototype, which are summarized in Table 7.4.

Table 7.4: List of Test Scenarios with Different Supply Chain Structures and Calculation Methods

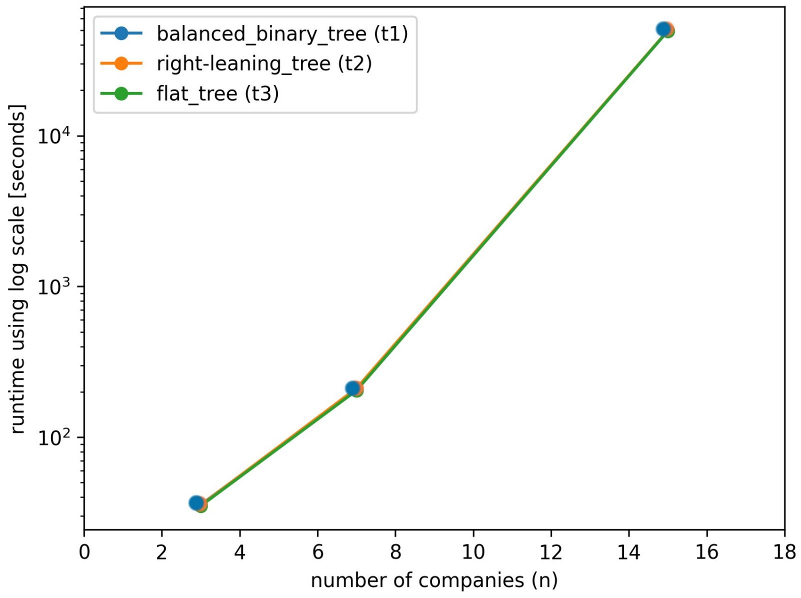
Supply Chain Structure	Calculation Method	Test Scenario
Balanced Binary Tree	Optimized Calculation	Balanced Binary Tree with Optimized Calculation
	Full Calculation	Balanced Binary Tree with Full Calculation
	Traditional Calculation	Balanced Binary Tree with Traditional Calculation
Right-leaning Tree	Optimized Calculation	Right-leaning Tree with Optimized Calculation
	Full Calculation	Right-leaning Tree with Full Calculation
	Traditional Calculation	Right-leaning Tree with Traditional Calculation
Flat Tree	Optimized Calculation	Flat Tree with Optimized Calculation
	Full Calculation	Flat Tree with Full Calculation
	Traditional Calculation	Flat Tree with Traditional Calculation

We have run these test scenarios for various sizes of the supply chains. The server we run our experiments on allows us to evaluate scenarios with up to 17 companies, without encountering any resource conflicts. As the *balanced binary tree* supply chain structure requires $2^m - 1$ companies, we could include the setup for 3, 7, and 15 companies. Since the *right-leaning tree* supply chain structure requires $2^m - 1$

companies, we could include the setup specifically tailored for an odd-numbered number of companies. Furthermore, we *excluded* test scenarios that would take longer than a day to run. The reported mean runtimes (t_i) are the average taken over three executions of the same scenario. For all scenarios, the quotient of the standard deviation and the mean ($\frac{\delta_i}{t_i}$) is at most around 0.06, which confirms our assumptions that the values used for the actual computation do not have a significant influence on the overall runtime. In other words, the runtime for an LCA depends mostly on the structure of the supply chain and the calculation method.

Experiment 1

The first question we would like to answer is the impact of the structures of supply chains on LCA computation using SMPC naively. Our hypothesis is that the structure has no significant impact on the performance of the LCA computation when using the direct application of SMPC. In order to test our hypothesis, we designed an experiment in which we execute LCA computations for our three supply chain scenarios of varying supply chain structure, and using the traditional LCA calculation method for various values of n. n represents the number of participating companies in a supply chain. The information regarding the experiment is presented in Table 7.5. The outcome of the experiment is presented in Figure 7.6.



n	t_1[h]	$\frac{\delta_1}{t_1}$	t_2[h]	$\frac{\delta_2}{t_2}$	t_3[h]	$\frac{\delta_3}{t_3}$
3	00:00:35	0.011	00:00:35	0.030	00:00:35	0.027
7	00:03:24	0.008	00:03:31	0.033	00:03:24	0.008
15	13:49:05	0.019	13:48:54	0.019	13:44:20	0.019

Figure 7.6: Runtimes of Supply Chain Scenarios with Traditional LCA Calculation Method

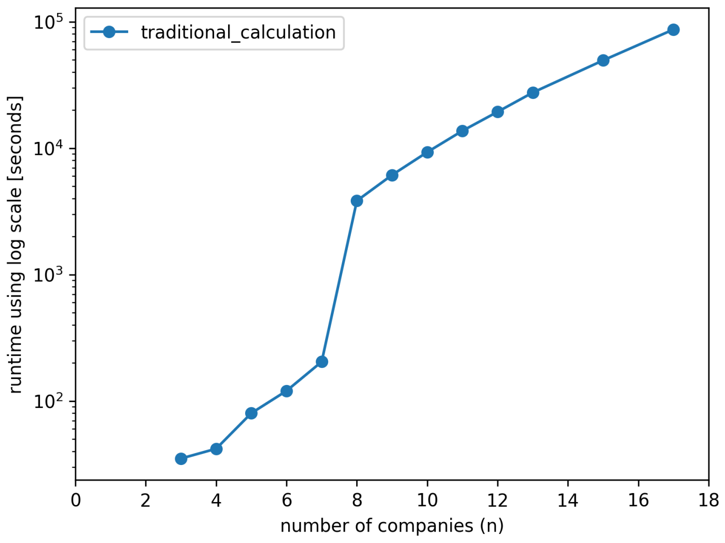
Since we excluded test scenarios that take longer than a day to run, we were able to run tests for supply chain scenarios with up to 17 participants for the traditional calculation method. Among these three supply chain scenarios, which vary in their structures, there exist only three shared instances, namely, tests involving 3, 7, and 15 companies. The results show that as the number of companies increases, the differences in runtime between the LCA scenarios with varying structures also increase, but only slightly. For instance, for 15 companies, the difference is merely 4 minutes and 45 seconds (0.5%) between maximum and minimum runtimes, which are 13 hours, 49 minutes and 5 seconds, and 13 hours, 44 minutes and 20 seconds, respectively. The test results demonstrate that the structures of supply chains have a negligible impact on the performance of LCA computations when the computations are performed with the traditional LCA calculation method using SMPC naively. Therefore, our hypothesis is verified.

Table 7.5: Impact of Supply Chain Structure on LCA Computation Using SMPC Naively

Experiment 1	
Question	How do the structures of supply chains impact the performance of LCA computation using SMPC naively?
Hypothesis	Supply chain structure has no significant impact on the performance of LCA computation when using the direct application of SMPC.
Experiment	We designed an experiment that executes LCA computation for three different supply chain scenarios with varying supply chain structures and using the traditional LCA calculation method for various numbers of companies.
Result	The structures of supply chains have negligible impact on the performance of LCA computations when the computations are performed with the traditional LCA calculation method using SMPC naively.
Conclusion	The hypothesis is verified.

Experiment 2

In our second experiment, described in Table 7.6, our goal is to determine the impact of the number of companies on the performance of LCA using SMPC naively. Our hypothesis is that the runtimes for LCA computation increase in a cubic way as the number of companies increases. The sizes of the matrices in the LCA computation vary based on the number of participating companies. Since matrix inversion is the most time-consuming operation in the computation, the runtimes for LCA computation will depend on the cubic behavior of matrix inversion as the company count increases. In our second experiment, we execute LCA computation for the flat supply chain using the traditional LCA calculation method for various numbers of companies. This experiment basically demonstrates the behavior of a direct application of SMPC to LCA.



n	t[h]	$\frac{\delta}{t}$
3	00:00:35	0.028
4	00:00:41	0.022
5	00:01:20	0.056
6	00:02:00	0.046
7	00:03:24	0.008
8	01:03:53	0.003
9	01:42:09	0.012
10	02:34:58	0.001
11	03:48:26	0.001
12	05:23:52	0.001
13	07:38:37	0.020
15	13:44:20	0.019
17	24:24:32	0.019

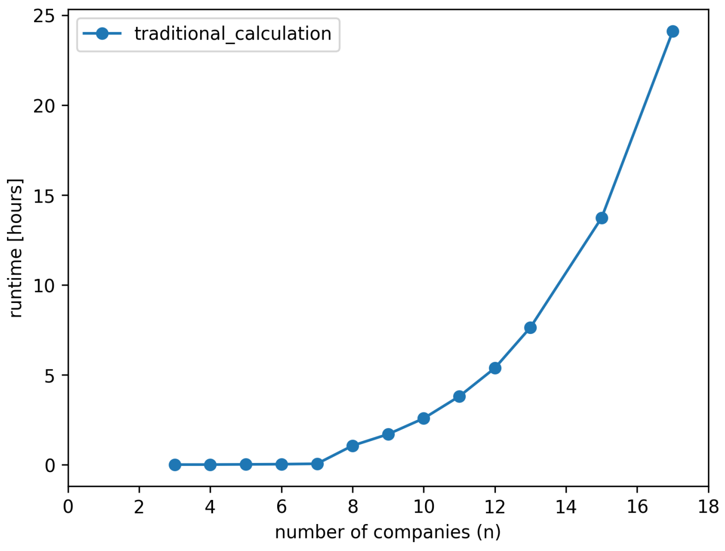


Figure 7.7: Runtimes of a Flat Tree with Traditional LCA Calculation Method

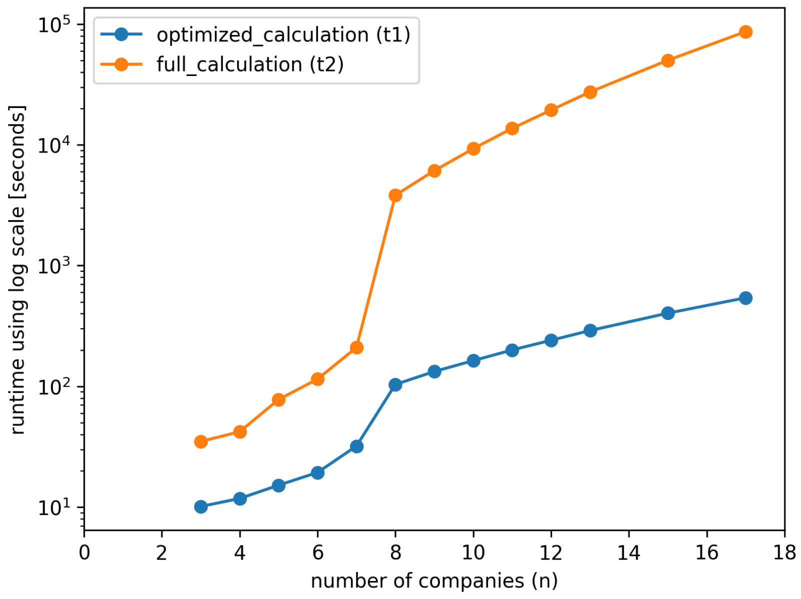
Figure 7.7 shows the runtime results of executing the flat supply chain using the traditional LCA calculation method for various numbers of companies. It highlights a negative correlation between the growth in the number of companies and the computation's performance, meaning that as the number of participating companies increases, the computation's runtime grows at a cubic rate. The significance of this relationship cannot be understated, as it has important implications for the efficient design and deployment of future systems in which a large number of entities are involved. It is essential to note that we should analyse scenarios with fewer than eight companies separately from those with more. The use of SMPC results in a significantly higher overhead in scenarios with more than eight companies. While scenarios with less than eight companies can be analysed in minutes, scenarios with more than eight companies can take several hours. The reason for the sudden increase in the runtime results between test scenarios involving 7 and 8 companies in the computation is explained in our previous traditional LCA experiments (Experiment 1), which can be found in Section 7.4. However, it is important to highlight that this sudden increase does not affect the main result of our work. The detailed discussion will be presented in Subsection 7.7.3. Furthermore, the growing overhead of SMPC as more companies are involved is not unique to the LCA scenario. This outcome could also be applicable to other scenarios.

Table 7.6: Impact of the Number of Participating Companies on LCA Computation Using SMPC Naively

Experiment 2	
Question	What is the impact of the number of companies on the performance of LCA using SMPC naively?
Hypothesis	The computational time for LCA increases in a cubic way as the number of companies increases.
Experiment	We designed an experiment that executes an LCA computation for the flat supply chain using the traditional LCA calculation method for various numbers of companies.
Result	As the number of participating companies increases, the computation runtime experiences growth at a cubic rate.
Conclusion	The hypothesis is verified.

Experiment 3

The number of companies involved in SMPC is not the only factor that affects the performance of scenarios. Both the number and complexity of operations performed using SMPC also play an important role. In experiment 3, we aim to evaluate the effectiveness of our approach in improving LCA computation performance for a group by minimizing the number of operations performed in SMPC. The details of the experiment can be found in Table 7.7. We hypothesize that reducing the number of complex operations performed in SMPC will result in improved computation performance due to a decrease in the complexity of the SMPC operations. Therefore, our hypothesis is that by excluding matrix inversion and scaling vector calculation operations, our approach improves performance significantly. We execute LCA computation for the flat supply chain using both optimized and full LCA calculation methods, with varying numbers of companies. In this experiment, the optimized LCA calculation demonstrates our approach, while the full LCA calculation represents the standard approach of LCA computation using SMPC directly, which means including all operations in SMPC.



n	t_1[h]	t_2[h]
3	00:00:10	00:00:34
4	00:00:11	00:00:42
5	00:00:15	00:01:17
6	00:00:19	00:01:54
7	00:00:31	00:03:29
8	00:01:43	01:03:50
9	00:02:12	01:41:58
10	00:02:43	02:34:58
11	00:03:19	03:48:31
12	00:04:00	05:23:52
13	00:04:49	07:37:47
15	00:06:43	13:57:46
17	00:09:01	24:07:26

Figure 7.8: Runtimes of a Flat Tree with Optimized and Full LCA Calculation Methods

Figure 7.8 illustrates the runtime results of executing the flat supply chain using both optimized and full LCA calculation methods, with varying numbers of companies, n. As the number and complexity of operations involved in SMPC reduces, the runtime results of the LCA computation reduce significantly, as illustrated in the figure. For instance, the runtime of the LCA computation for a supply chain involving 17 companies typically takes 24 hours when all operations are performed in SMPC. However, our approach, which reduces the number of complex operations in SMPC, results in a significantly reduced runtime of just nine minutes.

Table 7.7: Impact of Operation Complexity on SMPC Overhead

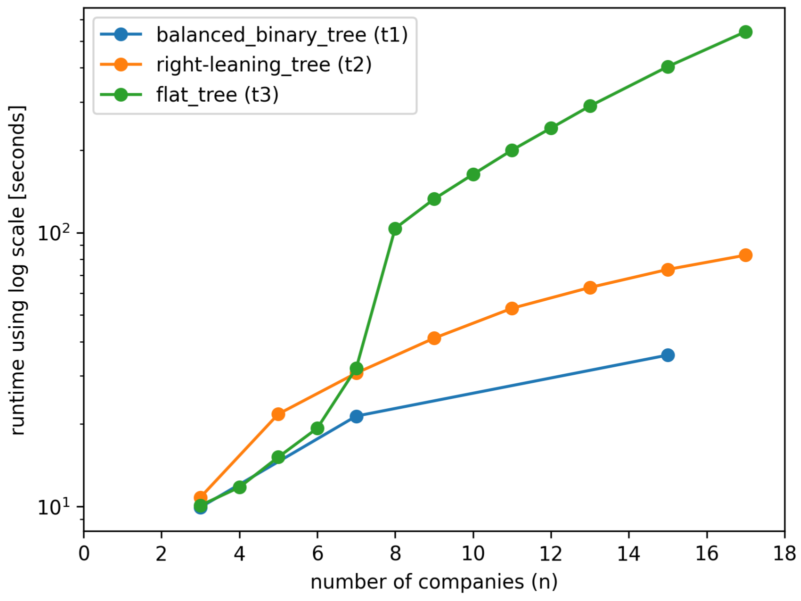
Experiment 3	
Question	What is the impact of our approach on the LCA computation for a group?
Hypothesis	By excluding matrix inversion and scaling vector calculation operations, our approach improves performance significantly.
Experiment	We designed an experiment that executes LCA computation for the flat supply chain using both optimized and full LCA calculation methods, with varying numbers of companies.
Result	Our approach leads to a significant decrease in the runtime of LCA computation.
Conclusion	The hypothesis is verified.

Experiment 4

In the previous experiments, we first evaluated the effect of supply chain structure on the performance of LCA computation using a direct application of SMPC. We then examined how increasing the number of companies involved in SMPC affects runtime and finally, demonstrated the impact of reducing the operations performed using SMPC on the runtime. Now, we proceed to examine the runtime results of various LCA test scenarios using our approach. We will illustrate the enhancement in performance achieved through our approach in LCA scenarios.

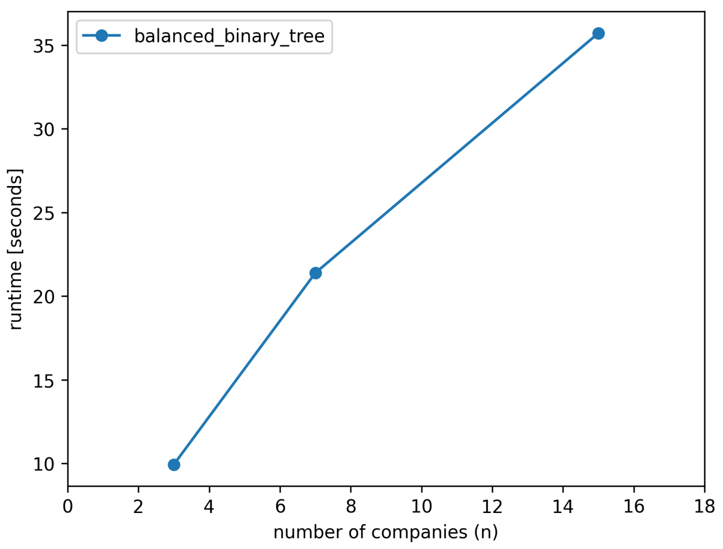
Supply chain scenarios can differ from one another in their structure. Our next experiment aims to determine the impact of our approach on LCA computation performance in various supply chain scenarios with differing structures. Our hypothesis is that our approach significantly enhances performance in scenarios where the supply chain structure allows for hierarchical groupings and parallel computations. To validate our hypothesis, we execute LCA computation for three supply chain scenarios, as outlined in Subsection 7.7.1, with varying supply chain structures. We apply our approach to perform LCA computations for these scenarios.

To evaluate the impact of our approach on various supply chain scenarios, we present a comparison of the runtime results for the test scenarios in Figure 7.9 and Figure 7.10. These figures show the results of executing the three supply chain scenarios with our approach, which is the optimized LCA calculation method. According to the results presented in the figures, the performance of computing the LCA in a supply chain using our approach is primarily determined by the structure of the supply chain.

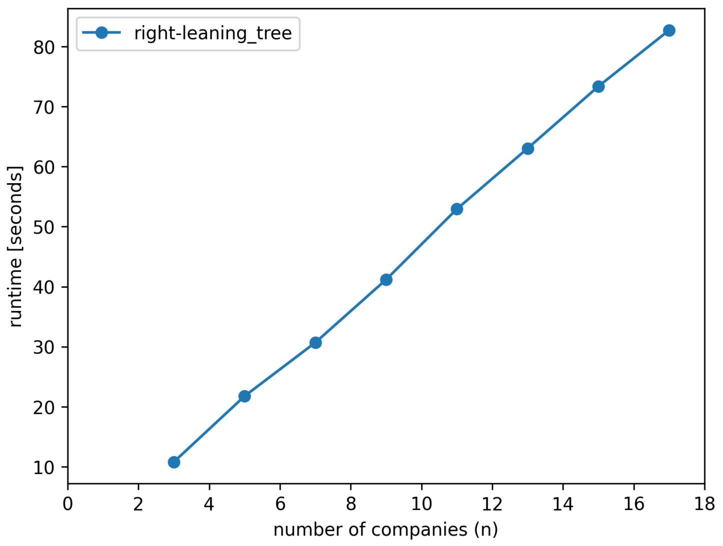


n	t_1[h]	t_2[h]	t_3[h]
3	00:00:09	00:00:10	00:00:10
4	-	-	00:00:11
5	-	00:00:21	00:00:15
6	-	-	00:00:19
7	00:00:21	00:00:30	00:00:31
8	-	-	00:01:43
9	-	00:00:41	00:02:12
10	-	-	00:02:43
11	-	00:00:52	00:03:19
12	-	-	00:04:00
13	-	00:01:03	00:04:49
15	00:00:35	00:01:13	00:06:43
17	-	00:01:22	00:09:01

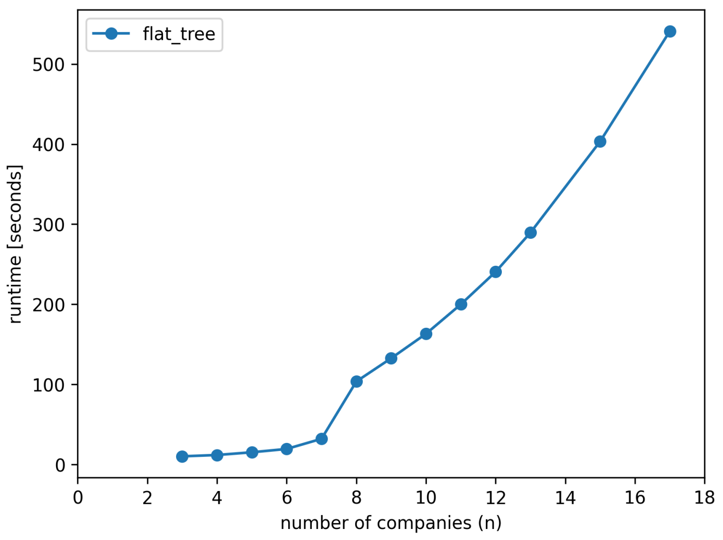
Figure 7.9: Runtimes of Supply Chain Scenarios with Optimized LCA Calculation Method



(a) Runtimes of Balanced Binary Tree Scenario



(b) Runtimes of Right-Leaning Tree Scenario



(c) Runtimes of Flat Tree Scenario

Figure 7.10: Runtimes of Supply Chain Scenarios with Optimized LCA Calculation Method

The results of the experiment show that:

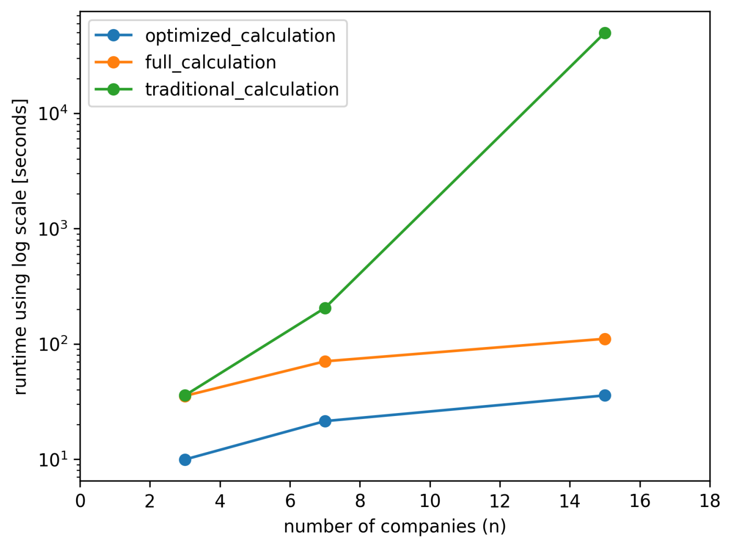
- The balanced binary tree scenario in our benchmarking involves hierarchical groups where LCAs for groups on the same level can be executed in parallel. As all groups have three members, the performance of LCA computation for the balanced binary tree supply chain scenario mainly depends on the number of groups that can be executed in parallel. As demonstrated in Figure 7.10a, the runtimes of our approach for the balanced binary tree increase approximately logarithmically with the number of companies in the supply chain. This is strong evidence that executing the LCA computations of the groups in parallel works well.
- The right-leaning tree scenario in our benchmarking also comprises hierarchical groups. However, the results of the LCA computations of the groups are interrelated, thus requiring sequential execution of all group computations. Given that each group has three members, the runtime is determined by the number of groups executed in sequence. As illustrated in Figure 7.10b, the runtimes of our approach for the right-leaning tree increase roughly linearly, as is expected for a supply chain whose length also grows linearly.
- The flat supply chain scenario in our benchmarking includes a single group that is the same size as the number of companies in the supply chain. As this scenario models a "flat" supply chain, its runtime is mostly determined by one large SMPC over all inputs from all partners of the supply chain. As seen from Figure 7.10c, the runtimes of our approach for the flat tree exhibit mostly polynomial growth.

Table 7.8: Impact of Our Approach on the Performance of LCA Computation of Different Supply Chain Scenarios

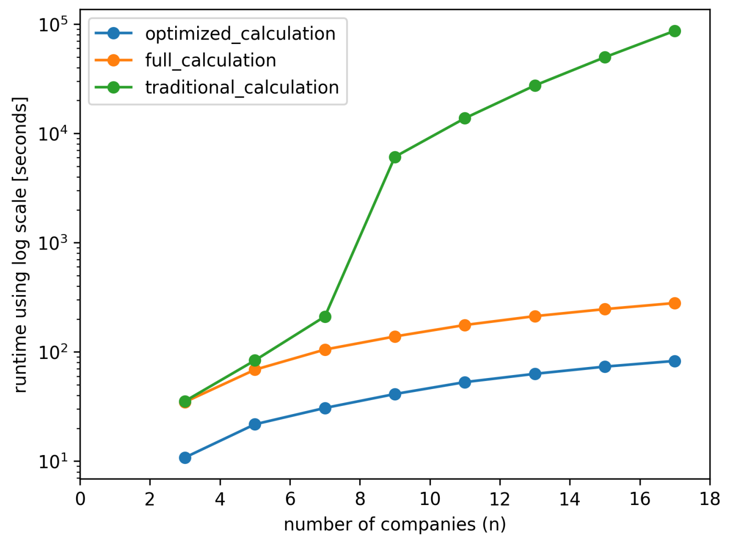
Experiment 4	
Question	How does our approach affect the performance of LCA computation in various supply chain scenarios with different supply chain structures?
Hypothesis	Our approach significantly enhances performance in scenarios where the supply chain structure allows for hierarchical groupings.
Experiment	We have designed an experiment that executes LCA computation for different supply chain scenarios with varying supply chain structures using our approach.
Result	The performance of LCA computation for a supply chain using our approach is primarily determined by the structure of the supply chain. For supply chains that are suitable for hierarchical grouping, our approach improves performance significantly.
Conclusion	The hypothesis is verified.

Experiment 5

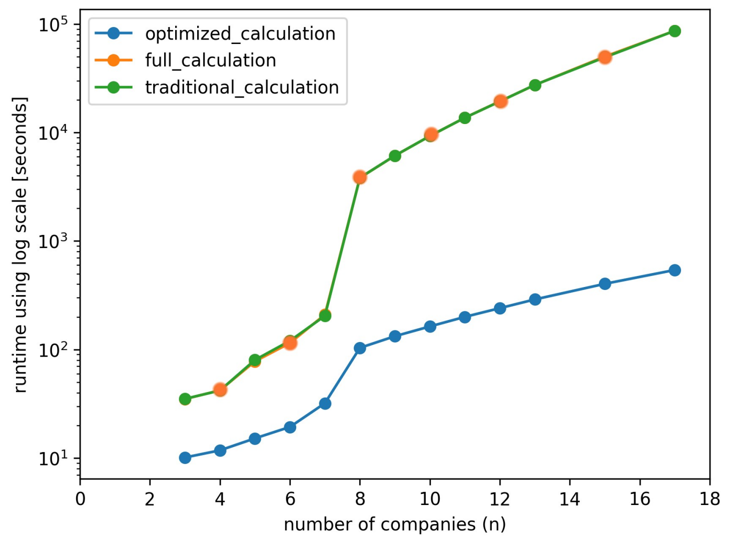
In our final experiment, we aim to compare the performance of the traditional LCA approach, which uses SMPC directly, with our confidentiality-enhanced LCA approach. The question we would like to address is whether our approach always provides better LCA computation performance for all supply chain scenarios as compared to the traditional LCA approach. We hypothesize that our approach improves the performance of LCA computation for all cases where the supply chain structure allows for hierarchical groupings. For other cases, it performs LCA computation at least as well as the traditional LCA method. To verify our hypothesis, we have designed an experiment executing LCA computation for the three supply chain scenarios with varying supply chain structures. The LCA results for these scenarios were determined using three different calculation methods for various numbers of companies. These methods include the optimized LCA calculation method, which represents our approach, the full LCA calculation method, and the traditional LCA calculation method.



(a) Runtimes of Balanced Binary Tree Scenario



(b) Runtimes of Right-Leaning Tree Scenario



(c) Runtimes of Flat Tree Scenario

Figure 7.11: Runtimes of Supply Chain Scenarios using Different LCA Calculation Methods

The diagrams in Figure 7.11 present the runtime results of this experiment. Each diagram includes the runtime results of test scenarios calculated using the different LCA calculation methods for a specific supply chain structure. The key takeaways from these diagrams are:

- One key feature of our scalable privacy-preserving model is the use of hierarchical grouping. The hierarchical grouping aspect of our approach enhances the performance of calculating LCA for supply chains by reducing the number of companies involved in SMPC. The *full LCA calculation method* shown in the figures demonstrates the hierarchical grouping aspect of our approach. While this method still performs all steps of LCA in SMPC, it does so with *smaller groups*, resulting in reduced computation times. Since the *flat tree scenario* involves a single large group, the hierarchical grouping aspect of our approach, represented by the full calculation, does not alter the performance of test scenarios for the flat tree supply chain. These results are equivalent to those obtained using the traditional LCA calculation method for the flat tree supply chain, as shown in Figure 7.11c. However, for the *balanced binary tree* and the *right-leaning tree* scenarios, having hierarchical grouping, represented by the full calculation, have significantly reduced runtimes for LCA computation. These supply chain structures are demonstrated in Figure 7.11a and Figure 7.11b. The most significant improvement can be seen when the supply chain has a well-designed hierarchy for grouping and the capability to parallelize computation within these groups, like the balanced binary tree scenario.
- In addition to reducing the number of companies involved in SMPC, hierarchical grouping can also help us to optimize and minimize the number of operations that must be performed in SMPC. The optimized LCA calculation method demonstrates how we can optimize the operations and reduce the time complexity of the operations performed in SMPC. *The optimized LCA calculation method represents the approach proposed in our work for calculating the LCA of supply chain scenarios.* As demonstrated by the results, our approach proves to be the most effective among the different methods evaluated for the given test scenarios. The data shows that our method outperforms the others in terms of runtime, providing a clear advantage in terms of performance.

Instead of entering the unit value of their products, companies may prefer to manage their own scaling factors. In this case, it is necessary to compute all the calculation steps of LCA using SMPC within hierarchical groups to maintain the confidentiality of data shared in groups, which corresponds to the full LCA calculation method. From the results of executing the flat tree supply chain scenario with the full LCA calculation method, illustrated in Figure 7.11c, we can conclude that our approach always performs at least as well as the direct application of SMPC.

As such a flat supply chain is a rare exception and companies may also prefer to enter the unit values of their product, we expect that our approach will result in a significant performance gain in most real cases.

Table 7.9: Comparison of Traditional LCA Approach and Our Approach on the Performance of LCA Computation in Different Supply Chain Scenarios

Experiment 5	
Question	Does our approach always provide better LCA computation performance for all supply chain scenarios as compared to the traditional LCA approach?
Hypothesis	Our approach improves the performance of LCA computation for most cases where the supply chain structure allows for hierarchical groupings and parallel computations. For other cases, it performs LCA computation at least as well as the traditional LCA method.
Experiment	We have designed an experiment that executes LCA computation for the three supply chain scenarios with varying supply chain structures. The LCA results for these scenarios were determined using three different calculation methods for various numbers of companies.
Result	Our approach significantly reduces the runtimes of LCA computation for supply chain scenarios allowing for hierarchical grouping, while in others – like the flat tree in our experiment – it performs at least as well as the traditional LCA method using SMPC naively.
Conclusion	The hypothesis is verified.

7.7.3 Experimental Runtime Discussion

When executing supply chain scenarios using the traditional LCA calculation method, for scenarios including more than 17 participant companies, the runtime results exceed a duration of 24 hours. The significant overhead associated with the traditional LCA calculation method resulted in the exclusion of test scenarios where the runtime exceeded 24 hours, due to time constraints. However, the selected sizes of supply chain scenarios in our experiments are sufficient for drawing our conclusions. Adding larger examples does not significantly contribute to what we have already learned. First, as the number of participants in a computation increases, the overhead caused by SMPC increases significantly (Experiment 1 in Section 7.4). Second, our approach improves the performance significantly for scenarios allowing hierarchical grouping. With hierarchical grouping, we reduce the number of participants and the number of communications which lead to enhanced performance and prevents significant SMPC overhead (Experiments 4 and 5 in Section 7.7).

Our focus is not to optimise SCALE-MAMBA or SMPC at its core. Our work builds upon the existing frameworks. Within SCALE-MAMBA, the protocols change depending on the values of n choose t, where n represents the number of participating parties, and t denotes the threshold value. SCALE-MAMBA has made the design decision to work with pre-computed data for secret sharing in scenarios where n choose t is a small value. In our specific case, this occurs when the group size is less than 8 (i.e., $n < 8$), and the threshold is set to $n/2 - 1$. However, in both scenarios, whether pre-computed or not, our approach improves the performance. When we specifically examine the benchmarking aspect where SCALE-MAMBA incorporates this optimisation, we observe that our approach has a significant performance improvement in scenarios involving hierarchical groups. This implies that even if SCALE-MAMBA's performance is enhanced for larger group sizes, we would expect that our method still delivers a significant performance improvement. In addition, since our work is on top of SCALE-MAMBA framework, when SCALE-MAMBA becomes faster, our approach also gets much faster.

The computational effort required for SMPC scales with the number of participants, a principle that holds true for all existing approaches. Likewise, network communication effort also increases as the number of participants grows. By our grouping approach, we significantly reduce the number of participants in groups on average, leading to faster computations and lower communication overhead. Additionally, we enable some group computations and communications to be executed in parallel, further improving the performance. Since these factors apply to all frameworks, we expect similar performance improvements when using other SMPC frameworks in our approach.

The other discussion we want to include is that our experiments do not evaluate the impact of different bandwidth and network latency scenarios. As we stated in the previous paragraph, with our grouping approach, in scenarios involving hierarchical groups, we reduce the number of participants within one group. This significantly reduces the need for network communications that the number of network packages are sent. By splitting the communication into smaller groups, we also keep the communication overhead lower. We expect this to have a positive impact, especially for networks with high latency. In such cases, our performance improvements will most likely become even larger because, in addition to lower computational overhead, we have fewer network packages that need to be sent. For more in-depth research on the influence of network parameters on the performance of an SMPC framework, we refer interested readers to [158]. However, it is important to note that this research was conducted on a different SMPC framework, FRESCO, and the results cannot be directly applied to SCALE-MAMBA.

7.8 Summary

In this chapter, we have carried out a comprehensive evaluation of the performance of our approach using life-cycle assessment scenarios. The main aim of this evaluation was to demonstrate how our approach provides better performance in collaborative business scenarios, as compared to an approach that relies on the direct application of SMPC. We first evaluated the impact of naively applying SMPC to the computation of LCA. The results of our evaluation revealed that the direct application of SMPC to the LCA computation resulted in a substantial decrease in the performance of the computation, due to the significant overhead incurred. We then evaluated the performance of our enhanced LCA model through a series of experiments. These experiments involved analysing LCA supply chain scenarios with different supply chain structures. From the results of the experiments, we concluded that our approach results in most cases in a significant performance gain, while in others, it performs at least as well as the traditional LCA approach with direct application of SMPC. While our focus in this chapter is on the LCA scenario, the performance evaluation presented herein would also be applicable to other collaboration scenarios beyond LCA.

8. Related Work

The work presented in this thesis has multiple steps involving and evaluating different aspects and perspectives to achieve the complete approach. One of the main contributions of the work presented in this thesis is to take and adapt existing works and ideas on threat modelling and SMPC and make them work together in a specific context. We apply them to problems in the business collaboration context. We have therefore grouped the related work chapter into two main sections which address different aspects of security and privacy in business collaborations:

- Addressing security and privacy issues
- Preserving security and privacy

8.1 Addressing Security and Privacy Issues

8.1.1 Security and Privacy in Business Collaborations

Contemporary IT systems operate in the context of complex multi-party collaborations where parties share their private information/data, resources, and responsibilities and compute collaborative computations to achieve organizational goals. In those systems, parties are no longer closed entities. Instead, they are actors that are engaged in processes of cooperation within complex networks. The paradigm of collaborative networks is a promising approach to providing solutions to some challenges, such as shortening innovation cycles or increased competition due to the globalized world and the current economic and financial crisis that companies face today. The entities in those collaborative networks are largely autonomous, geographically distributed, and have heterogeneous infrastructures [159]. Such networks provide benefits for many different sizes of organizations. For example, small and medium-sized entities are generally unable to provide their services and products directly to customers due to financial and geographical restrictions, although they might be experts and innovative in specific areas. These networks allow these small and medium-sized entities to collaborate with other organizations to achieve their goals. In addition, larger organizations might want to provide their services on a global scale (entering new markets). They too can benefit from acting in collaborative networks [93].

Although collaborative business networks have indisputable benefits, the increasing interdependence of the participating companies causes additional risks and challenges regarding information security and privacy, as well as highly disruptive impacts [93, 94]. The essential characteristic of collaborative business networks is to share resources in terms of information, data, or services with other organizations in the network and to access them accordingly. Companies can be reluctant to join collaborative business networks because most of the information in those networks is shared and easily accessible within the network. Therefore, it is essential to conduct security and privacy analyses of such collaborative business networks to minimize the unnecessary disclosure of any confidential information, to encourage parties to participate in the networks.

There have been some papers studying security and privacy problems in a collaborative business network context and addressing them from different points of view. Wohlgemuth *et al.* [94] introduce an approach to achieving acceptable secure business networking applications despite threats due to covert channels. They adapt resilience to the implementation of IT security in business networking applications. Gogoulos *et al.* [160] design a privacy-aware decision engine operating within synergistic contexts. There are some other papers that discuss the security and privacy issues and/or threats in collaborative business networks in specific scenarios, such as industrial IoT-based applications [10], smart grids [161], and blockchain applications [162]. All these papers are essential to being aware of problems and provide solutions regarding security and privacy problems in collaborative business networks. However, they are either scenario-specific or they contain limited resources to give a complete understanding or an approach that analyses and solves the security and privacy problems in collaborative business networks.

8.1.2 Threat Modelling

In order to address security and privacy problems and conduct security and privacy analyses of systems, threat modelling is a stepping stone. There are a wide range of threat modelling approaches, e.g. [101, 103, 163, 164, 165], that differ in the set of threats covered or the way threats are modelled. The most widely used threat model is STRIDE [101], which categorizes different types of threats and simplifies the overall security conversation. STRIDE covers the main six broad categories of threats, which are Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. STRIDE elicits threat scenarios and, subsequently, derives security use cases. There are some threat modelling techniques that use the STRIDE model as a basis, like fuzzy logic and the SDL threat modelling tool. The fuzzy logic-based threat modelling [166] method is based on fuzzy set theory and uses STRIDE to determine the input variables. The SDL threat modelling tool [167, 168] provides automated analysis of security threats of systems and is based on the STRIDE model (it can be represented by data flow diagrams (DFDs)).

There are other types of threat modelling approaches. For example, T-MAP [169] is a quantitative threat modelling method that calculates the total severity weights of attack paths relevant to information technology systems and quantifies security threats. On the other hand, Attack Trees [170] represent attacks against a system using a tree structure with the goal at the root node and different ways of achieving that goal at the leaf nodes. Both approaches have the support of an automated tool; Tiramisu for T-MAP and SecureITree for Attack Trees, respectively. Also, there are some graphical threat modelling and specification languages used to analyse and communicate risk scenarios and model system risks and security, such as CORAS [171], ArchiMate's risk and security overlay [172], and the bowtie method [173]. For modelling threats and undesirable behaviours, they provide customized use case diagrams.

All the approaches given above mostly focus on eliciting potential security threats; they do not cover privacy threats. Threat modelling for privacy issues is an emergent and important area. Therefore, some approaches have also been developed to cover potential privacy threats. One of the most popular approaches is the LINDDUN methodology. LINDDUN is a systemic approach for privacy threat modelling to elicit the privacy requirements of software-intensive systems and decide privacy-enhancing technologies accordingly, and it is a mirror of STRIDE. It stands for the violations of seven privacy properties: linkability, identifiability, non-repudiation, detectability, disclosure of information, content unawareness, policy and consent noncompliance. It includes a process, threats, and a requirements discovery method. The other approaches for privacy issues are privacy considerations for internet protocols (PCIP) and privacy impact assessments (PIAs). The PCIP approach [174] outlines a set of combined security-privacy threats and a set of privacy-specific threats and presents a list of mitigations and some general guidelines for protocol designers, while privacy impact assessments (PIAs) [175] offer a way for organizations to detect potential privacy issues, take countermeasures, and build customised protections before making huge investments in the development of a new technology, service, or product. Also, there are studies in the literature to conceptualize privacy. We refer the reader to [165, 176] for a comprehensive understanding of privacy.

The closest related works to our threat modelling approach are STRIDE [101] and LINDDUN [103]. With STRIDE, we share the threat categories focusing on traditional security properties (e.g. spoofing, tampering) and with LINDDUN the privacy-focused threat categories (e.g. linkability, identifiability). To provide a holistic security and privacy assessment, we used STRIDE and LINDDUN as starting points for developing our threat categories. In contrast to STRIDE and LINDDUN, we are not aiming for a generic threat modelling approach that is applicable to a wide range of application domains. Our aim is to develop a threat modelling approach that is tailored to inter-organizational collaboration scenarios.

8.1.3 Threat Modelling for Business Collaborations

There are some studies providing systematic threat modelling for collaborative business networks, but they focus on specific business collaboration scenarios. The focus of their threat modelling varies; the focus can be software-centric [28, 29], asset-centric [30], or threat-centric. In addition, some studies [29, 30] use existing well-defined threat modelling methodologies for the security analysis in their system when implementing their threat modelling. Sabbagh and Kowalski [28] build a socio-technical framework for threat modelling of software supply chain systems. The focus of their threat modelling is software, and the framework provides a thinking aid for identifying social and technical threats to software supply chains and describing appropriate countermeasures. Nevertheless, as they also state, although the framework helps to identify the security problems in software supply chains, the countermeasures to mitigate these problems are not specified. Also, there is another work [29] that identifies and analyses cyber threats to the cyber supply chain (CSC) domain and models attacks using the widely known STIX threat model. This is an elegant threat modelling approach that focuses on technical threats in software being built or in a system being deployed. They claim that ensuring regular third-party auditing and security policy implementation in line with international standards by all stakeholders is a required action to provide CSC controls and proper mitigations. Syed et al. [30] provide an extensive threat modelling report on supply chain traceability systems. Their threat modelling approach is systematically asset-centric, and they adopt the STRIDE threat model to describe common threats in supply chains. Their focus is on technology and computer systems; they do not analyse the additional vulnerabilities and threats that can be introduced by people and business processes. There are also other studies presenting and/or assessing security threats and risks in specific scenarios of business collaboration, but they do not have systematic threat modelling [177, 178].

Moreover, it is well-known that inter-enterprise collaborations can create significant security or privacy risks [15, 93, 94, 95]. This fact stimulated research in extending BPMN with support for security or privacy requirements (e.g. [179, 180, 181, 182, 183]), monitoring security or privacy policies at runtime (e.g. [184, 185]), and integrating security or privacy support into business-process or decision-engines (e.g. [160, 182]). We consider our threat modelling approach complementary to these approaches. By using our threat modelling approach, the efforts required for adding additional security or privacy measures can be focused on protecting the most vulnerable assets. In addition, there are several frameworks (e.g. [186, 187, 188]) for assessing the security and privacy properties of business processes. For example, Anica [188] allows systems to be assessed with respect to their security levels while focusing on privacy aspects; [186] present a privacy-aware business process modelling platform for inferring and enforcing privacy constraints; and PLEAK [187] is a tool for analysing privacy-enhanced BPMN models. Also, there are some graphical modelling languages such as CORAS [171], ArchiMate's Risk and Security Overlay [172], and The Bowtie Method [173], that are used to analyse and communicate risk scenarios and model enterprise risks and security.

Last but not least, there is a study motivating and justifying that our work is relevant. That study [189] focuses on a stakeholder collaboration paradigm for modelling security requirements of IT systems. Although this study has a different perspective from our approach, it has conceptional similarities with our approach in that we need to take relations and tensions between the various system parties into consideration to understand and identify security risks to a system.

In our threat modelling approach, the focus is mostly on the things that a business would like to protect. This model focuses on the collaborative parts of business networks that are not controlled by laws or industry regulations; it helps companies understand their risks in order to decide how much they would like to collaborate or how they can share their data while at the same time being able to control their risks. In this way, they can make informed decisions about the benefits and potential risks of sharing information and collaboration. We open up business networks to allow for more collaborations, more information/service sharing, and more joint computations while still empowering companies to control what they share and to understand what potential risks are. If they are willing to accept those risks, they can collaborate. If they do not accept those risks, then they do not collaborate in that way.

8.2 Preserving Security and Privacy

8.2.1 Security and Privacy in Business Process Modelling

In settings of business networks, including the collaborations of two or more organizations/entities, improvements in a process can be achieved by sharing and utilizing information across interested organizations. Traditional inter-organizational business process approaches tend to offer a standardized process that applies uniformly to all participating organizations. As a result, these approaches tend to overlook the unique characteristics and differences in authority levels and perspectives among the participating entities. This could limit the effectiveness of the process and hinder the achievement of the intended goals [190]. Therefore, organizations/entities are reluctant to share details of their processes and/or require significant countermeasures to prevent the disclosure of their confidential information, such as valuable trade secrets, in any (external) data processing [191].

From the business domain, there are several related works, e.g. showing that concerns about confidentiality and lack of trust are one of the main reasons preventing close collaborations in business processes [15, 18, 19, 95]. In order to provide a diversity of participating organizations and protect their business privacy, the concept of the process view has been proposed by [192] and various studies have contributed to this concept [193, 194, 195, 196]. Adopting process views gives organizations better control over their privacy and confidentiality. However, such advantages come at the expense of complexity in process representation, a great deal of view updating, and other things [190]. With the wide adoption of the internet, collaborative business processes (CBPs) are executed via collaborative networks in which autonomous, geographically distributed, and heterogeneous organizations collaborate to achieve common goals [4]. The most common solution for these collaborative networks is to use cloud computing because of its benefits. It dynamically provides infrastructure, platform, and software resources from any location in the world over the internet, offers ready-to-consume IT services, increases efficiency, reduces cost and time-to-market, and so on [12].

8.2.2 Privacy-Enhancing Technologies

Differential Privacy

Differential privacy (DP) is a statistical anonymity model that allows information about individuals to be collected without compromising their privacy. It deals with the paradox that it is possible to gain important information about a population while learning nothing about an individual [197]. The primary objective of DP is to guarantee that each individual data item in the data set is provided the same level of privacy regardless of whether or not the observation is included in the data set [198]. There are several application domains that adopt and implement DP to preserve data privacy, such as statistical databases, mining sensitive data, big data, blockchain, machine learning, and IoT-enabled critical infrastructures. We refer the reader to [199] for details.

Differential privacy includes a process of adding the desired amount of tailored noise into data sets and altering identifiable information within whole data sets by using various mathematical algorithms. When this process is applied to large amounts of data, it becomes a provable guarantee of privacy. There are noise addition mechanisms, also referred to as data perturbation mechanisms, that add noise to the data to preserve data privacy [200]. The amount of noise included in the data set has a direct correlation with the sensitivity value, Δf, and privacy loss, ϵ [201]. For DP, there are three noise addition mechanisms, which are Laplace, Exponential, and Gaussian mechanisms, respectively. Deciding which mechanism is used in a system depends on the operations and queries required in the system.

The strengths of DP that sustain its application for a variety of uses can briefly be gathered under two aspects: protection against linkage attacks and measurement of privacy loss. In most cases, DP guarantees the neutralization of linking attacks and eliminates the risks of re-identification [198, 202]. Also, DP allows for control over the amount of information leakage permitted while maintaining data utility [198]. Although DP has strengths, it is important to highlight that it does not promise complete privacy preservation: it suffers from three major limitations [203, 204]. First, it is difficult to achieve DP during large query sensitivity while keeping the desired statistical properties required for precise inference. Second, it is challenging to provide useful data for application scenarios where multiple queries are required on the data. Third, there can be an uncertainty of outcome: since differentially private mechanisms tend to generate results that differ considerably, it can cause a decrease in reliability [199].

Homomorphic Encryption

Homomorphic encryption (HE) is a form of encryption that allows a third party (e.g. cloud, service provider) to carry out specific types of computations on the encrypted data instead of on the actual data itself. The desirable feature of HE is that during the computations, the features of computation functions and the format of the encrypted data are preserved: it enables complex mathematical operations to be carried out on encrypted data without compromising the encryption [100, 205]. There are several significant applications of HE: electronic voting applications [206, 207], the domain of control and optimization [208], big data [209, 210], cloud computing [211, 212, 213, 214, 215], private use of untrusted web servers [216], protection of mobile agents [217], and more. We refer the reader to [205, 218] for more details.

According to the number of operations that can be performed on the encrypted data, HE schemes are mainly classified into three broad categories as follows: partially homomorphic encryption (PHE), somewhat homomorphic encryption (SWHE), and fully homomorphic encryption (FHE). PHE allows only one type of operation to be performed, but this operation can be executed an unlimited number of times. SWHE enables some types of operations with a limited number of times. On the other hand, FHE enables an unlimited number of operations to be carried out an unlimited number of times [100].

Although homomorphic encryption has a desirable feature, which is the main advantage of the scheme, that it can perform operations on cipher text, it has some important limitations. First of all, typically, homomorphic ciphers do not offer verified computing: in order to verify the correctness, another scheme is needed. Also, PHE and SWHE schemes are generally too limited in functionality for most uses. In addition to this, although FHE enables an unlimited number of operations on encrypted data, because of the limitations on efficiency of FHE schemes (extremely slow and computationally expensive), their use is still not feasible in all real-life applications [100, 205]. Even if a really effective FHE were to be discovered in the future, problems would still remain. For example, there is no support for multiple users. There can be users in a system that wish to protect their data from the provider. In order to support this, one possible solution would be to provide a separate database for every user, encrypted under that user's public key. However, this solution would become infeasible quickly with a large database and many users. There is a promising approach to address this problem, which is defining and constructing multi-key FHE. Nevertheless, this problem is still open to being proven (one issue is the circular security of FHE). For details, we refer the reader to [100, 219].

Secure Multi-Party Computation

Secure multi-party computation (SMPC) is another of the most commonly used privacy-enhancing technologies. Since we implement SMPC in our approach, we provided a detailed explanation of SMPC in Section 2.3. However, we discuss the limitations of SMPC in this chapter for a proper comparison between the technologies. It is important to highlight that SMPC is an important tool, but SMPC applications are effective for specific problems (functions). In addition to this, although SMPC has two important requirements – confidentiality and correctness – it also has limitations in both requirements. First of all, it indicates that private input data is protected during the execution of the protocol. However, there is no statement about what can be inferred from seeing the output of the computation. Also, in order to keep the input data private, SMPC needs to make assumptions about the number of malicious coordinating participants in the computation. For example, in our average salary example (in Subsection 2.3.3), if Bob and Alice collude, they would be able to learn John's salary. Another limitation is correctness. The function that participants set out to compute cannot be changed because it is encrypted (thus limiting the ability of actual participants to change the computation), but they can still enter an incorrect value because any input is allowed in SMPC [64]. SMPC has limitations not only in security and confidentiality but also in performance. SMPC has a significant communication overhead that affects the performance of computations to a considerable extent. SMPC is still too slow to be used in naïve and direct way. In Chapter 4, we explain in detail how we integrate SMPC into our approach and eliminate some of its limitations.

8.2.3 Integration of Privacy-Enhancing Technologies in Business Processes

There are a considerable number of existing works that integrate privacy-enhancing technologies into the execution of business processes (e.g. [96, 97, 98, 99, 220]). Similar to our work, they present domain-specific adaptions of these technologies to solve a particular information-sharing problem. Brandon Kuczenski et al. [98] propose an approach for life-cycle assessment (LCA) that provides a shared secure computation (aggregation) model. The proposed model enables computations on private input data without requiring disclosure to any party. The model works with two common third parties: a secure aggregator and a secure co-processor. The system runs a joint function between participants (including all companies in the supply chain and the third parties, aggregator and co-processor) in order to prevent the disclosure of companies' confidential information and to make secure computations. For privacy concerns, they use an additional homomorphic encryption scheme for secure aggregation. However, this encryption scheme requires fixed-point arithmetic; the capacity for privacy-preserving life cycle impact assessment (LCIA) is significantly constrained by the fixed-point requirement. The model is suitable for carrying out any computation that can be represented as a set of weighted sums that contains aggregation problems, such as horizontal averaging or vertical aggregation.

Florian Kerschbaum et al. [99] present a secure software-as-a-service called Sustainability Benchmarking Service (SBS) to solve the information-sharing problem. The service has two important security objectives: providing confidentiality during processing and preventing information disclosure from the reports, respectively. In order to protect the data during processing, the service uses Paillier's encryption scheme, which is additively homomorphic encryption that allows the addition of plaintexts using the ciphertexts only. Also, differential privacy (Laplacian noise) is used to prevent leakages from the reports. Although the service gives some benefits, the technologies used have limitations. For example, additive homomorphic encryption sets bounds to the system. Besides this, adding noise to data can reduce the data utility in the system. If the data includes sensitive information, the system can require precise inference. It can be difficult to achieve this requirement with DP.

One of today's hottest technologies, blockchain, has also been considered for use in support of inter-organizational processes. Although using blockchain to deploy inter-organizational processes can bring a number of advantages, such as trust, transparency, and accountability, it also brings additional security issues. For example, blockchain, by its very nature, does not have primitives to guarantee data confidentiality. There are also other issues, such as data integrity, the confidentiality of the process, and trust in the correct execution of the process [221]. To address some of these, blockchain exploits various privacy-preserving methods: zero-knowledge protocol [222, 223], secure multi-party computation [224, 225], and homomorphic encryption [226]. Although these solutions provide a good starting point, they are not enough. These solutions have been developed initially to address the issue that blockchain data is publicly available (e.g. transactions, smart contracts). They do not, however, address the problems that an inter-organizational process, by its collaborative nature, intrinsically implies [221].

Recently, another hot research topic, federated learning (FL), was introduced to improve AI applications. The core idea of federated learning (also known as collaborative learning) is to train machine learning models on separate datasets which are held by multiple decentralized devices/servers or parties while preserving the privacy of those datasets to a certain extent. As federated learning allows participants to exchange knowledge without disclosing private information, it is a very promising method for enhancing decentralised machine learning models. However, the characteristics and requirements of industrial clients are not taken into account by many existing FL approaches. Therefore, FL is still not suitable for the industrial setting [227, 228]. Nevertheless, there is a quite new work [229] that proposes an FL system with a process description and software architecture to offer FL as a service (FLaaS) to industrial clients deployed to edge devices. Although the proposed industrial federated learning (IFL) system is an important step towards incorporating FL in industrial applications, there are open problems, such as client volatility, model performance, and resource-optimized training on heterogeneous edge devices. Beyond the industry concept, secure multi-party computation has been widely used in many privacy-preserved federated learning (PPFL) methods [230, 231, 232, 233]. However, the main challenge of SMPC-based FL methods is to improve computational efficiency. Completing a training round in SMPC-based FL frameworks requires high computational power and substantial computational resources [234].

In addition, there are quite a number of studies that work on preserving the privacy of multi-party computations/collaborations in the IoT-fog-cloud ecosystem by applying privacy enhancing technologies [235, 236, 237, 238, 239]. Zheng and Cai [235] developed a privacy-preserved data sharing framework that can be used for industrial internet of things (IIoT) scenarios where multiple parties are included in different stages of the system. In order to preserve sensitive information for parties, they adopt differential privacy (DF). Lyu et al. [236] propose an efficient and privacy-preserving aggregation system for smart grid applications with the help of fog computing architecture that allows the intermediate fog nodes to periodically collect data from nearby smart meters (SMs) and accurately derive the aggregation of the fine-grained fog-level aggregation for further cloud-level aggregation without disclosing any information from the connected smart meters. They use a Gaussian mechanism to guarantee differential privacy of the aggregate statistics. Also, to mitigate privacy loss and maintain the utility of the data, they use a two-layer encryption scheme. Li et al. [238] propose a lightweight privacy-preserving approach based on homomorphic encryption in the context of IIoT. Their aim is to shift the computation costs from resource-constrained IoT devices to a third-party powerful server while ensuring privacy protection for and between data owners, third-party cloud servers, and data users. Although the given approaches provide some good security and privacy properties, they still rely on centralized systems and/or third-parties.

In order to ensure the data privacy of systems without the need for a third party and/or a centre, secure multi-part computation is one of the most suitable approaches. However, because of the limitations of SMPC regarding computational efficiency (performance and resource consumption), SMPC is not yet widespread in practice; it has limitations for its application in resource-constrained environments [64, 71]. One of the closest related works in literature that shares a similar idea with our approach is presented by Lopez-Fenner et al. [240]. They propose a privacy-preserved multi-party computation architecture for the IoT using fog computing. Their aim is to provide efficient data processing for applications in the IoT-fog-cloud ecosystem in a privacy-preserving context. In order to provide secure and privacy-preserving data processing at the edge, they take into consideration the fog and edge computing paradigms together with a multi-party computation method. In their study, they briefly discuss how this can be applied in a scenario like the present time of pandemic. However, they do not provide a formalization of their proposal, and they do not provide a robust evaluation of the efficiency and limitations of their proposal.

In our privacy-preserving approach, the focus is to provide a scalable privacy-preserving model for business collaborations that minimizes the amount of confidential data that needs to be shared between participants in business collaborations while, at the same time, improving performance. Our approach is based on a combination of decomposing collaborations in business networks and the use of secure multi-party computation. We use SMPC to provide increased protection of data required for collaborations. However, as we stated in the previous subsection (Subsection 8.2.3), SMPC is still too slow to use in a naïve and direct way. Therefore, we decompose business collaborations based on their relationships in the networks, and we then adopt SMPC for each of those decomposed groups. We believe that this combination can enable closer collaborations within businesses. With our approach, we open up business networks to allow secure collaborations without the need for a third party or a centre while still ensuring the performance efficiency of the collaborations.

8.2.4 General-Purpose Compilers for SMPC

There are various general-purpose compilers available for SMPC, offering high-level abstractions to define arbitrary functions and execute secure computation protocols. Some notable frameworks include ABY [241], EMP-toolkit [242], MP-SPDZ [243], MPyC [244], Obliv-C [245], ObliVM [246], PICCO [247], SCALE-MAMBA [248], and Sharemind [249]. Some works ([243, 250]) have explored these frameworks and evaluated them based on different criteria, such as language expressiveness, cryptographic back-end capabilities, and developer accessibility. The findings are summarized in Table 8.1 and Table 8.2. These tables primarily rely on [250]. However, it also incorporates supplementary data sourced directly from the publications of the frameworks, which is not covered in [250]. It is important to acknowledge that each framework has its own strengths and emphasizes different aspects of SMPC, such as efficiency, security, ease of programming, or specific application domains. The selection of a framework depends on the specific requirements of the application and the trade-offs that need to be considered.

Table 8.1: A Summary of Defining Features and Documentation Types of Frameworks (mainly adopted from [250])

Frameworks	Protocol family	Parties supported	Mixed -mode	SH	Mal	DH Maj	Language docs	Online support	Example code	Example docs	Open source
ABY	GC, MC	2	●	●	○	-	●	○	●	●	●
EMP-toolkit	GC	2	●	●	●	-	○	○	●	○	●
MP-SPDZ	GC, MC Hybrid	2+	◑	●	●	●	○	●	●	●	●
MPyC	MC	1+	●	●	○	○	●	○	●	●	●
Obliv-C	GC	2	●	●	○	-	●	○	●	◑	●
ObliVM	GC	2	●	●	○	-	○	○	●	○	●
PICCO	Hybrid	3+	●	●	○	○	●	○	◑	○	●
SCALE-MAMBA	Hybrid	2+	●	●	●	●	●	●	●	◑	●
Sharemind	Hybrid	3	●	●	○	○	●	●	●	●	◑

SH: Semi-honest, Mal: Malicious (with abort), DH Maj: Dishonest majority; ○ no support at all, ◑ partial support, ● full support

Table 8.2: A Summary of Functionality and Expressibility of Each High-Level Language (mainly adopted from [250])

(a) Data Types

Frameworks	Boolean	Fixed int	Arbitrary int	Float	Array	Dynamic array	Struct
ABY	◑	●	○	◑	●	○	●
EMP-toolkit	●	●	●	●	●	●	●
MP-SPDZ	●	●	●	●	●	○	●
MPyC	●	●	●	●	●	-	-
Obliv-C	●	●	○	●	●	●	●
ObliVM	○	●	●	●	◑	●	◑
PICCO	◑	●	●	●	●	●	●
SCALE-MAMBA	○	●	◑	●	●	○	◑
Sharemind	●	○	●	●	●	●	●

(b) Operators

Frameworks	Logical	Comparisons	Addition	Multiplication	Divison	Bit -shifts	Bitwise
ABY	●	●	●	●	○	○	○
EMP-toolkit	●	●	●	●	●	●	●
MP-SPDZ	●	●	●	●	●	●	●
MPyC	●	●	●	●	●	●	●
Obliv-C	●	●	●	●	●	●	●
ObliVM	●	●	●	●	●	●	●
PICCO	●	●	●	●	●	●	●
SCALE-MAMBA	○	●	●	●	●	●	●
Sharemind	●	●	●	●	●	●	●

○ no support at all, ◑ partial support, ● full support

For our work, there are two essential features that we require in a framework: the ability to support an arbitrary number of parties and robust security guarantees. Collaborative business scenarios typically involve multiple participants. Therefore, the chosen framework for SMPC must be capable of accommodating an arbitrary number of parties. Furthermore, as we aim to apply SMPC to real-life scenarios, strong security measures are necessary. A reliable source, [250], provides clear recommendations for frameworks and essentially classifies SCALE-MAMBA as a good “all-around” solution. Notably, it aligns perfectly with our must-have features, supporting computations involving two or more parties and defending against various adversaries, including malicious entities with a dishonest majority [248]. Additionally, it is well-documented and uses an easy-to-learn high-level language, making it a good choice. Consequently, we have decided to use SCALE-MAMBA framework to assess our work.

The decision to use SCALE-MAMBA was made during the early stages of the Ph.D. Presently, there is a more advanced alternative called MP-SPDZ, which offers better performance and supports a wider range of data types (e.g., boolean) and operators (e.g., logical) [243]. However, at the time of selecting the framework, the early version of MP-SPDZ was at a comparable level [251, 252]. Importantly, while MP-SPDZ would be a more advanced choice, it would not make any difference in the value of our contributions. Regardless of which framework we use, our contributions (findings and results) would remain the same. A significant factor in our work is the number of participants in a group. With the approach presented in this thesis, we reduce the number of participants significantly on average, leading to faster computations. This reduces the number of communications in a group, which means less communication overhead. Also, we have an additional aspect that data is definitely being shared with a smaller number of external parties, thereby improving security. In addition, our approach simplifies the entire management process and increases resilience. For instance, in the direct application of SMPC, computation is blocked as soon as one party fails to provide input, whether intentionally or due to network outages. However, with our approach of distributing parties into smaller groups and conducting local computations, the impact of one party not reacting is significantly reduced. It becomes a localized issue instead of a global one. These factors remain the same and do not depend on the underlying SMPC framework. With a faster framework, computations would generally be faster, but the only change would be in the actual performance measurement. However, the goal of performance measurement in our work is to compare the performance of our approach with the direct implementation of SMPC. To make this comparison meaningful, we use the same framework for both cases. Regardless of the framework chosen, the general performance improvement that we achieved stays the same.

9. Conclusion and Future Work

9.1 Conclusion

In this thesis, we have seen multiple contributions that were directed towards the common goal of enhancing the security and confidentiality of sensitive information in inter-organizational collaborations while minimizing any negative impact on performance. We have achieved this by using our scalable privacy-preserving model, which is based on a combination of decomposing computations in collaborations and the use of secure multi-party computation (SMPC). With our model, we have enabled confidential computations on shared data within collaborative business networks *without the need for a trusted third party.* In addition, we have enhanced the performance of business networks using SMPC, making it a suitable choice for real-world applications.

In more detail, in this study, we started by developing a novel threat modelling approach for inter-organizational collaborations. On the one hand, our threat modelling allows companies to better understand the risk of joining such a collaboration. On the other hand, it can be used to design and develop secure and privacy-friendly inter-organizational software systems. Using our threat modelling approach, we were able to identify the potential threats for companies associated with joining such collaborations and provide systematic guidance for creating strategies and designing secure and privacy-friendly business networks. These are presented in Chapter 3. Using the potential threats and security and privacy requirements identified through our threat modelling approach, we have built a scalable privacy-preserving model for inter-organizational collaborations. Through the hierarchical grouping approach in our model, we have minimized the sensitive data being shared between companies in collaborations. Additionally, we have eliminated the requirement for a trusted third party by using SMPC within these hierarchical groups. This is discussed further in Chapter 4 and Chapter 5.

As previously stated, our model can be applied to real-world scenarios. To further demonstrate this, in Chapter 6 we have conducted a case study, based on life-cycle assesment, using our approach. We have developed a confidentiality-enhanced life-cycle assessment model based on our approach. Alongside the abstract security guarantees provided by SMPC, our confidentiality-enhanced LCA model has provided additional security properties to ensure the confidentiality of the sensitive information of the participants during the LCA process. We have also developed a prototype for the case study. With the prototype, we were able to evaluate the effectiveness of our approach in the case study. The evaluation results of the enhanced LCA model in Chapter 7 show that our model provides a significant performance improvement in scenarios involving hierarchical structures, which allow the decomposition of computations within the business networks compared to the direct application of SMPC to them. We believe that since our approach minimizes the data being shared between participants of the supply chain while, at the same time, improving LCA model performance, it can enable closer collaborations within a supply chain, particularly by enabling precise and real-time LCAs to determine the environmental impact of products from supply chains. It is important to note that our approach and the prototype we have developed can also be adapted for use in other business network scenarios, particularly those with a hierarchical structure. Cloud computing, IoT, and Industry 4.0 are transformative technologies that have enabled inter-organizational collaborations in many ways. Our approach can be used to improve the privacy and security of data in inter-organizational collaborations in cloud computing, IoT, and Industry 4.0, such as collaborative product development [253], predictive maintenance [254], and smart grid management [255].

9.2 Future Work

We see several avenues of future work. First of all, SMPC ensures the privacy of the input data during the computation and guarantees that the output of the computation is accurate, even if some of the parties involved are malicious or malfunctioning. Nevertheless, there is still the possibility that participants may enter inaccurate values. To overcome this issue, we plan to integrate our model with commitment schemes, which will enable us to verify that all participants input the correct values to computations. There are various commitment schemes [256] that can be used with SMPC, including Pedersen commitment [257, 258], Damgard–Jurik commitment [259] and Oblivious Transfer [71]. These schemes allow multiple parties to commit to a value without revealing the value to the other parties, while still allowing the parties to verify the correctness of the committed value at a later time. It is important to note that the choice of commitment scheme will depend on the specific requirements of the SMPC protocol and the security level that is needed.

Second, there is another promising area with regards to extending the scope of our approach. We plan to develop a security and privacy analysis that goes beyond the rather abstract security guarantees provided by SMPC. SMPC guarantees that participants of a computation only know their own inputs and the result of the computation, but it ignores the information that participants of a collaboration can infer from information learned within one or several business collaborations.

We plan to extend our threat analysis to include such inferred information, further supporting companies in their decision to join (or not join) a collaboration. As we would like to have collaborative business networks that make multiple computations and analyses without violating privacy requirements and prevent the disclosure of more information than is intentionally shared, we need to formulate privacy goals of our approach and describe their analyses. Therefore, we plan to extend our approach by focusing on the formal privacy analyses of our model. There is a quite new approach called (α, β)-privacy [260], introduced in 2019, to specify privacy goals in a novel and declarative way. However, in this approach, there is only an example set of operations for cryptography (encryption, decryption, signing, and pair) at present. We plan to analyse the privacy goals of privacy-enhancing techniques using this approach and integrate them into our model. With this, we will be able to verify the privacy goals of our model with formal privacy analysis.

Furthermore, identifying all threats can be an incredibly tedious process involving extensive analysis, investigation, and continuous monitoring of various systems and data sources. Also, the outcomes can differ based on user experiences and perspectives. Their understanding of security risks, and awareness of emerging threats can influence the accuracy and effectiveness of the identification process [261]. Using artificial intelligence (AI) shows significant potential in addressing these obstacles and enhancing the process of identifying threats [262, 263]. AI technologies, such as machine learning and natural language processing, can analyze vast amounts of data at incredible speeds, assisting security analysts in identifying potential threats more efficiently and accurately [261, 263, 264]. By learning from historical data, AI algorithms can recognize known threat patterns and adapt to new and emerging ones, enabling proactive threat detection [265]. Additionally, AI-powered systems can integrate with external threat intelligence feeds and databases, continuously updating their knowledge about the evolving threat landscape [264]. This integration enables them to identify and respond to emerging threats promptly. By automating certain aspects of the threat identification process, AI can relieve security professionals of repetitive and time-consuming tasks, allowing them to focus on more complex analysis and decision-making [265]. As part of our future work, we aim to improve and automate the initial phase of threat identification within our threat modeling process through the utilisation of AI. However, it is important to note that human expertise remains crucial in the collaborative effort with AI [266]. Therefore, we would like to emphasize that our goal is not to replace human involvement but rather to leverage AI technology as a powerful tool to augment and streamline the threat modeling process.

In addition to that, we also are considering the integration of our threat modelling approach into a threat modelling tool, e.g., OWASP Threat Dragon [267]. This should make the threat modelling process more systematic and user-friendly for business experts who are, usually, not security or privacy experts. A further interesting line of research is the integration of guidelines for selection security measures, e.g. for privacy-enhancing technologies [268].

As the approaches and models proposed in this thesis are multidisciplinary, our future work also aims to enhance and validate them by incorporating research methodologies from other related disciplines. For instance, there exist several business modelling methodologies and techniques that can be used to validate our threat modelling approach. These methodologies can help to ensure that the approach effectively identifies, assesses, and mitigates potential threats and risks to the business. Examples include SWOT analysis [269, 270], Risk Assessment Frameworks (e.g., NIST Cybersecurity Framework, ISO 27001), Regulatory Compliance (e.g., GDPR, HIPAA), and feedback loop. However, it is important to note that threat modelling is an ongoing process, and validation methods may evolve over time. The ultimate goal is to ensure that the approach remains aligned with the changing needs of the organization and the evolving threat landscape.

Lastly, in inter-organizational collaborations, when parties often come together to share information, they also jointly produce design documents and blueprints. Similarly, there may be situations where parties might need to share design information with external parties, such as suppliers or clients to get feedback or manufacturing assistance. However, such data can contain sensitive information that requires protection from unauthorized access. SMPC can offer a solution by allowing the secure processing of this information while safeguarding the parties' intellectual property from potential attackers [271]. While SMPC was initially developed for numerical computations, it has been extended to work with text data, enabling secure string matching, pattern searching, and other text-related computations [80, 243, 247, 272, 273]. As part of our future work, we intend to explore the application of SMPC in enhancing the security of design documentation and blueprint within business collaborations.

Bibliography

[1] E. N. Loukis, N. Kyriakou, K. Pazalos and S. Popa. ‘Inter-organizational innovation and cloud computing’. In: *Electronic Commerce Research* 17.3 (2017), pp. 379–401 (Cited on pages 15 and 21).

[2] C. Hardy, N. Phillips and T. B. Lawrence. ‘Resources, knowledge and influence: The organizational effects of interorganizational collaboration’. In: *Journal of management studies* 40.2 (2003), pp. 321–347 (Cited on page 15).

[3] E. N. Sochneva, A. A. Malakhova, O. V. Starova, D. V. Zyablikov and D. I. Kravtsov. ‘Collaborations in the Modern Economy’. In: *Frontier Information Technology and Systems Research in Cooperative Economics*. Ed. by A. V. Bogoviz, A. E. Suglobov, A. N. Maloletko, O. V. Kaurova and S. V. Lobova. Springer International Publishing, 2021, pp. 63–71. ISBN: 978-3-030-57831-2 (Cited on page 15).

[4] D. Cocconi, J. Roa and P. D. Villarreal. ‘Cloud-based platform for collaborative business process management’. In: *Proceedings of Latin American Computer Conference*. Córdoba, Argentina: IEEE, 2017, pp. 1–10 (Cited on pages 15 and 169).

[5] W. Z. Khan, M. Rehman, H. M. Zangoti, M. K. Afzal, N. Armi and K. Salah. ‘Industrial internet of things: Recent advances, enabling technologies and open challenges’. In: *Computers & Electrical Engineering* 81 (2020). Article no. 106522 (Cited on page 16).

[6] M. Skilton and F. Hovsepian. *The 4th Industrial Revolution*. Firts. Palgrave Macmillan Cham, 2018 (Cited on page 16).

[7] F. Schoenthaler, D. Augenstein and T. Karle. ‘Design and governance of collaborative business processes in industry 4.0’. In: *Proceedings of the Workshop on Cross-organizational and Cross-company BPM co-located with the 17th IEEE Conference on Business Informatics*. 2015, p. 7 (Cited on page 16).

[8] A. Khanna and S. Kaur. ‘Internet of Things (IoT), Applications and Challenges: A Comprehensive Review’. In: *Wireless Personal Communications* 114.2 (2020), pp. 1687–1762 (Cited on page 16).

[9] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal and B. Sikdar. ‘A survey on IoT security: application areas, security threats, and solution architectures’. In: *IEEE Access* 7 (2019), pp. 82721–82743 (Cited on page 16).

[10] J. Pennekamp, M. Dahlmanns, L. Gleim, S. Decker and K. Wehrle. ‘Security considerations for collaborations in an industrial IoT-based lab of labs’. In: *Proceedings of IEEE Global Conference on Internet of Things*. IEEE. 2019, pp. 1–7 (Cited on pages 16 and 165).

[11] S. Hackett. *The Transformative Impact of the Cloud*. Tech. rep. 2016, pp. 1–13 (Cited on page 16).

[12] M. R. M. Ardakani, S. M. Hashemi and M. Razzazi. ‘A Cloud-based solution/reference architecture for establishing collaborative networked organizations’. In: *Journal of Intelligent Manufacturing* 30.5 (2019), pp. 2273–2289 (Cited on pages 16 and 169).

[13] P. Hofmann and D. Woods. ‘Cloud Computing: The Limits of Public Clouds for Business Applications’. In: *IEEE Internet Computing* 14.6 (2010), pp. 90–93 (Cited on page 17).

[14] C. Colicchia, A. Creazza, C. Noè and F. Strozzi. ‘Information sharing in supply chains: a review of risks and opportunities using the systematic literature network analysis (SLNA)’. In: *Supply Chain Management: An International Journal* 24 (1 2019), pp. 5–21 (Cited on page 17).

[15] T. T. H. Tran, P. Childerhouse and E. Deakins. ‘Supply chain information sharing: challenges and risk mitigation strategies’. In: *Journal of Manufacturing Technology Management* 27.8 (2016), pp. 1102–1126 (Cited on pages 17, 40, 71, 167 and 168).

[16] M. Javaid, A. Haleem, R. P. Singh, S. Rab and R. Suman. ‘Significance of sensors for industry 4.0: Roles, capabilities, and applications’. In: *Sensors International* 2 (2021), p. 100110. ISSN: 2666-3511 (Cited on page 17).

[17] M. Z. Babai, J. E. Boylan and B. R. Tabar. ‘Demand forecasting in supply chains: a review of aggregation and hierarchical approaches’. In: *Int. J. Prod. Res.* 60.1 (2022), pp. 324–348 (Cited on page 17).

[18] C. Sahin, B. Kuczenski, Ö. Egecioglu and A. E. Abbadi. ‘Towards Practical Privacy-Preserving Life Cycle Assessment Computations’. In: *Proceedings of the 7th ACM on Conference on Data and Application Security and Privacy*. ACM, 2017, pp. 167–169 (Cited on pages 17, 71, 102 and 168).

[19] P. Arbuckle, E. Kahn and A. Kriesberg. ‘Challenge Paper: Challenges to Sharing Data and Models for Life Cycle Assessment’. In: *Journal of Data and Information Quality* 9.1 (2017). Article no. 6 (Cited on pages 17, 71, 101, 102 and 168).

[20] A. C. Yao. ‘Protocols for Secure Computations (Extended Abstract)’. In: *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. Chicago, Illinois, USA: IEEE Computer Society, 1982, pp. 160–164 (Cited on pages 17 and 59).

[21] D. Evans, V. Kolesnikov and M. Rosulek. ‘A Pragmatic Introduction to Secure Multi-Party Computation’. In: *Foundations and Trends in Privacy and Security* 2.2-3 (2018), pp. 70–246 (Cited on pages 17, 40 and 118).

[22] M. von Maltitz and G. Carle. ‘A Performance and Resource Consumption Assessment of Secret Sharing Based Secure Multiparty Computation’. In: *Proceedings of Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Vol. 11025. Lecture Notes in Computer Science. Barcelona, Spain: Springer, 2018, pp. 357–372 (Cited on page 17).

[23] D. Falamas and K. Marton. ‘Performance Impact Analysis of Rounds and Amounts of Communication in Secure Multiparty Computation Based on Secret Sharing’. In: *Proceedings of the 18th RoEduNet Conference: Networking in Education and Research*. Galati, Romania: IEEE, 2019, pp. 1–6 (Cited on page 17).

[24] J. Kölsch, C. Heinz, A. Ratzke and C. Grimm. ‘Simulation-Based Performance Validation of Homomorphic Encryption Algorithms in the Internet of Things’. In: *Future Internet* 11.10 (2019). Article no. 218 (Cited on page 17).

[25] B. Alaya, L. Laouamer and N. Msilini. ‘Homomorphic encryption systems statement: Trends and challenges’. In: *Computer Science Review* 36 (2020). Article no. 100235 (Cited on page 17).

[26] M. T. Al-Nory. ‘Optimal Decision Guidance for the Electricity Supply Chain Integration With Renewable Energy: Aligning Smart Cities Research With Sustainable Development Goals’. In: *IEEE Access* 7 (2019), pp. 74996–75006 (Cited on page 17).

[27] K. Katsaliaki, N. Mustafee, S. J. E. Taylor and S. C. Brailsford. ‘Comparing conventional and distributed approaches to simulation in a complex supply-chain health system’. In: *J. Oper. Res. Soc.* 60.1 (2009), pp. 43–51 (Cited on page 17).

[28] B. Al Sabbagh and S. Kowalski. ‘A socio-technical framework for threat modeling a software supply chain’. In: *IEEE Security & Privacy* 13.4 (2015), pp. 30–39 (Cited on pages 17, 40 and 167).

[29] A. Yeboah-Ofori and S. Islam. ‘Cyber security threat modeling for supply chain organizational environments’. In: *Future Internet* 11.3 (2019). Article no. 63 (Cited on pages 18, 40 and 167).

[30] N. F. Syed, S. W. Shah, R. Trujillo-Rasua and R. Doss. ‘Traceability in supply chains: A Cyber security analysis’. In: *Computers & Security* 112 (2022). Article no. 102536 (Cited on pages 18 and 167).

[31] A. D. Brucker and S. Yalman. ‘Confidentiality Enhanced Life-Cycle Assessment’. In: *Proceedings of International Conference on Business Process Management*. Vol. 436. Lecture Notes in Business Information Processing. Rome, Italy: Springer, 2022, pp. 434–446 (Cited on pages 20 and 40).

[32] L. M. Carneiro, A. L. Soares, R. Patrício, A. L. Azevedo and J. P. de Sousa. ‘Case studies on collaboration, technology and performance factors in business networks’. In: *International Journal of Computer Integrated Manufacturing* 26.1-2 (2013), pp. 101–116 (Cited on page 21).

[33] C. Brindley. *Supply chain risk*. First. Routledge, 2004 (Cited on page 22).

[34] M. Christopher. ‘Logistics and Supply Chain Management: Strategies for Reducing Cost and Improving Service’. In: *International Journal of Logistics Research and Applications* (1999) (Cited on page 22).

[35] K. L. Croxton, S. J. Garcia-Dastugue, D. M. Lambert and D. S. Rogers. ‘The supply chain management processes’. In: *The International Journal of Logistics Management* 12.2 (2001), pp. 13–36 (Cited on page 22).

[36] L. M. Ellram. ‘Supply-chain management: the industrial organisation perspective’. In: *International Journal of Physical Distribution & Logistics Management* 21.1 (1991), pp. 13–22 (Cited on page 22).

[37] M. Weber, M. Hiete, L. Lauer and O. Rentz. ‘Low cost country sourcing and its effects on the total cost of ownership structure for a medical devices manufacturer’. In: *Journal of Purchasing and Supply Management* 16.1 (2010), pp. 4–16 (Cited on page 22).

[38] V. C. Pandey, S. K. Garg and R. Shankar. ‘Impact of information sharing on competitive strength of Indian manufacturing enterprises: An empirical study’. In: *Business Process Management Journal* 16.2 (2010), pp. 226–243 (Cited on page 22).

[39] C. Droge, J. Jayaram and S. K. Vickery. ‘The effects of internal versus external integration practices on time-based performance and overall firm performance’. In: *Journal of Operations Management* 22.6 (2004), pp. 557–573 (Cited on page 22).

[40] M. Christopher and U. Jüttner. ‘Developing strategic partnerships in the supply chain: a practitioner perspective’. In: *European Journal of Purchasing & Supply Management* 6.2 (2000), pp. 117–127 (Cited on page 22).

[41] C. Droge, S. K. Vickery and M. A. Jacobs. ‘Does supply chain integration mediate the relationships between product/process strategy and service performance? An empirical study’. In: *International Journal of Production Economics* 137.2 (2012), pp. 250–262 (Cited on page 22).

[42] H. Stadtler, C. Kilger and H. Meyr. *Supply chain management and advanced planning: concepts, models, software, and case studies.* Fifth. Springer Berlin, Heidelberg, 2015 (Cited on page 22).

[43] R. S. Kumar and S Pugazhendhi. ‘Information sharing in supply chains: An overview’. In: *Procedia Engineering* 38 (2012), pp. 2147–2154 (Cited on pages 22, 23 and 24).

[44] Z. Lotfi, M. Mukhtar, S. Sahran and A. T. Zadeh. ‘Information sharing in supply chain management’. In: *Procedia Technology* 11 (2013), pp. 298–304 (Cited on pages 22 and 23).

[45] H. L. Lee, K. C. So and C. S. Tang. ‘The value of information sharing in a two-level supply chain’. In: *Management Science* 46.5 (2000), pp. 626–643 (Cited on page 23).

[46] M. Barratt and R. Barratt. ‘Exploring internal and external supply chain linkages: Evidence from the field’. In: *Journal of Operations Management* 29.5 (2011), pp. 514–528 (Cited on page 23).

[47] H. Zhou and W. Benton Jr. ‘Supply chain practice and information sharing’. In: *Journal of Operations Management* 25.6 (2007), pp. 1348–1365 (Cited on page 23).

[48] S. Li and B. Lin. ‘Accessing information sharing and information quality in supply chain management’. In: *Decision Support Systems* 42.3 (2006), pp. 1641–1656 (Cited on page 23).

[49] F. Sahin and E. P. Robinson Jr. ‘Information sharing and coordination in make-to-order supply chains’. In: *Journal of Operations Management* 23.6 (2005), pp. 579–598 (Cited on page 23).

[50] C. Chandra, J. Grabis and A. Tumanyan. ‘Problem taxonomy: a step towards effective information sharing in supply chain management’. In: *International Journal of Production Research* 45.11 (2007), pp. 2507–2544 (Cited on page 23).

[51] Z. Yu, H. Yan and T. C. E. Cheng. ‘Benefits of information sharing with supply chain partnerships’. In: *Industrial Management & Data Systems* 101.3 (2001), pp. 114–121 (Cited on page 23).

[52] H. L. Lee and S. Whang. ‘E-business and supply chain integration’. In: *The Practice of Supply Chain Management: Where Theory and Application Converge.* Springer, 2004, pp. 123–138 (Cited on page 23).

[53] K. Inderfurth, A. Sadrieh and G. Voigt. 'The impact of information sharing on supply chain performance under asymmetric information'. In: *Production and Operations Management* 22.2 (2013), pp. 410–425 (Cited on page 23).

[54] M. Khurana, P Mishra and A Singh. 'Barriers to information sharing in supply chain of manufacturing industries'. In: *International Journal of Manufacturing Systems* 1.1 (2011), pp. 9–29 (Cited on page 23).

[55] H. L. Lee and S. Whang. 'Information sharing in a supply chain'. In: *International Journal of Manufacturing Technology and Management* 1.1 (2000), pp. 79–93 (Cited on pages 23 and 24).

[56] A. Ardichvili, V. Page and T. Wentling. 'Motivation and barriers to participation in virtual knowledge-sharing communities of practice'. In: *Journal of Knowledge Management* 7.1 (2003), pp. 64–77 (Cited on page 24).

[57] B. Bhargava, R. Ranchal and L. B. Othmane. 'Secure information sharing in digital supply chains'. In: *Proceedings of the 3rd IEEE International Advance Computing Conference*. IEEE. 2013, pp. 1636–1640 (Cited on page 24).

[58] D. Trcek. *Managing information systems security and privacy*. Springer Berlin, Heidelberg, 2006 (Cited on page 24).

[59] C. Adams and S. Lloyd. *Understanding PKI: Concepts, Standards, and Deployment Considerations*. 2nd. Addison-Wesley Longman Publishing Co., Inc., 2002 (Cited on pages 24 and 26).

[60] J. R. Vacca. *Public key infrastructure: building trusted applications and Web services*. First. Auerbach Publications, 2004 (Cited on page 24).

[61] A. Albarqi, E. Alzaid, F. Al Ghamdi, S. Asiri, J. Kar et al. 'Public key infrastructure: A survey'. In: *Journal of Information Security* 6.1 (2014), pp. 31–37 (Cited on page 24).

[62] S. Mazaher and P. Røe. 'A survey of state of the art in public key infrastructure'. In: *Norway, Norsk Regnesentral* (2003) (Cited on page 24).

[63] L. E. Hughes. 'X.509 Digital Certificate'. In: *Pro Active Directory Certificate Services: Creating and Managing Digital Certificates for Use in Microsoft Networks*. Apress, 2022, pp. 45–73 (Cited on page 25).

[64] Y. Lindell. 'Secure multiparty computation'. In: *Communications of the ACM* 64.1 (2021), pp. 86–96 (Cited on pages 28, 29, 31, 32, 171 and 174).

[65] C. Zhao, S. Zhao, M. Zhao, Z. Chen, C. Gao, H. Li and Y. Tan. 'Secure Multi-Party Computation: Theory, practice and applications'. In: *Information Sciences* 476 (2019), pp. 357–372 (Cited on pages 28, 29, 30 and 33).

[66] *Sharemind*. URL: `https://sharemind.cyber.ee` (Cited on page 28).

[67] *Duality*. URL: `https://duality.cloud` (Cited on page 28).

[68] *Unbound Tech.* URL: `www.unboundtech.com` (Cited on page 28).

[69] *Curv.* URL: `www.curv.co` (Cited on page 28).

[70] M. Ion, B. Kreuter, E. Nergiz, S. Patel, S. Saxena, K. Seth, D. Shanahan and M. Yung. *Private Intersection-Sum Protocol with Applications to Attributing Aggregate Ad Conversions.* Cryptology ePrint Archive, Paper 2017/738. 2017 (Cited on page 28).

[71] D. Evans, V. Kolesnikov and M. Rosulek. 'A Pragmatic Introduction to Secure Multi-Party Computation'. In: *Foundations and Trends in Privacy and Security* 2.2-3 (2018), pp. 70–246 (Cited on pages 28, 30, 32, 33, 174 and 178).

[72] R. Canetti. 'Security and Composition of Multiparty Cryptographic Protocols'. In: *Journal of Cryptology* 13.1 (2000), pp. 143–202 (Cited on page 29).

[73] Y. Aumann and Y. Lindell. 'Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries'. In: *Journal of Cryptology* 23.2 (2010), pp. 281–343 (Cited on page 30).

[74] R. Cramer, I. B. Damgård and J. B. Nielsen. *Secure Multiparty Computation and Secret Sharing.* First. Cambridge University Press, 2015 (Cited on page 31).

[75] O. Goldreich, S. Micali and A. Wigderson. 'How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority'. In: *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*. Ed. by A. V. Aho. ACM, 1987, pp. 218–229 (Cited on page 32).

[76] A. Wigderson, M. Or and S Goldwasser. 'Completeness theorems for noncryptographic fault-tolerant distributed computations'. In: *Proceedings of the 20th Annual Symposium on the Theory of Computing (STOC'88).* 1988, pp. 1–10 (Cited on page 32).

[77] D. Chaum, C. Crépeau and I. Damgard. 'Multiparty unconditionally secure protocols'. In: *Proceedings of the twentieth annual ACM symposium on Theory of computing.* 1988, pp. 11–19 (Cited on page 32).

[78] T. Rabin and M. Ben-Or. 'Verifiable secret sharing and multiparty protocols with honest majority'. In: *Proceedings of the twenty-first annual ACM symposium on Theory of computing.* 1989, pp. 73–85 (Cited on page 32).

[79] A. C.-C. Yao. 'How to generate and exchange secrets'. In: *27th Annual Symposium on Foundations of Computer Science (sfcs 1986).* 1986, pp. 162–167 (Cited on page 32).

[80] A. C. Yao. ‘How to Generate and Exchange Secrets (Extended Abstract)’. In: *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*. Toronto, Canada: IEEE Computer Society, 1986, pp. 162–167 (Cited on pages 33 and 180).

[81] D. W. Archer, D. Bogdanov, Y. Lindell, L. Kamm, K. Nielsen, J. I. Pagter, N. P. Smart and R. N. Wright. ‘From Keys to Databases - Real-World Applications of Secure Multi-Party Computation’. In: *The Computer Journal* 61.12 (2018), pp. 1749–1771 (Cited on page 33).

[82] R. Bendlin, I. Damgård, C. Orlandi and S. Zakarias. ‘Semi-homomorphic Encryption and Multiparty Computation’. In: *Proceedings of Advances in Cryptology – Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Vol. 6632. Lecture Notes in Computer Science. Tallinn, Estonia: Springer, 2011, pp. 169–188 (Cited on page 33).

[83] I. Damgård, V. Pastro, N. P. Smart and S. Zakarias. ‘Multiparty Computation from Somewhat Homomorphic Encryption’. In: *Proceedings of Advances in Cryptology - Annual Cryptology Conference*. Vol. 7417. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, 2012, pp. 643–662 (Cited on page 33).

[84] *Secure Order Matching*. URL: `https://partisia.com/better-market-solutions/order-matching/` (Cited on page 33).

[85] *Sepior Advanced MPC*. URL: `https://sepior.com/advanced-mpc-by-sepior` (Cited on page 33).

[86] F. Meisel, W. Rei, M. Gendreau and C. Bierwirth. ‘Designing supply networks under maximum customer order lead times’. In: *IIE Transactions* 48.10 (2016), pp. 921–937 (Cited on page 35).

[87] K. Raagul Srinivasan and J JawagarShrehari. ‘Applied Procedures for Lead Time Reduction: A Review’. In: *International Journal of Engineering Trends and Technology* 43 (2017), pp. 169–172 (Cited on page 35).

[88] A. G. Robinson and J. H. Bookbinder. ‘NAFTA supply chains: facilities location and logistics’. In: *International Transactions in Operational Research* 14.2 (2007), pp. 179–199 (Cited on page 35).

[89] J. M. Bossert and S. P. Willems. ‘A periodic-review modeling approach for guaranteed service supply chains’. In: *Interfaces* 37.5 (2007), pp. 420–436 (Cited on page 35).

[90] M. L. F. Cheong, R. Bhatnagar and S. C. Graves. ‘Logistics Network Design with Differentiated Delivery Lead Time: A Chemical Industry Case Study’. In: *Research Collection School Of Computing and Information Systems* (2004), pp. 1–20 (Cited on page 35).

[91] M. Paquet, A. Martel and B. Montreuil. *Manufacturing network design with reliable promising capabilities*. Tech. rep. Working Paper DT-2006-AM-2, 2006 (Cited on page 35).

[92] S. C. Graves and S. P. Willems. ‘Optimizing the supply chain configuration for new products’. In: *Management Science* 51.8 (2005), pp. 1165–1180 (Cited on page 35).

[93] C. Broser, C. Fritsch and O. Gmelch. ‘Towards Information Security Management in Collaborative Networks’. In: *Proceedings of Workshops on Database and Expert Systems Applications*. IEEE Computer Society, 2010, pp. 359–363 (Cited on pages 40, 164, 165 and 167).

[94] S. Wohlgemuth, S. Sackmann, N. Sonehara and A. M. Tjoa. ‘Security and privacy in business networking’. In: *Electronic Markets* 24.2 (2014), pp. 81–88 (Cited on pages 40, 165 and 167).

[95] T. T. Huong Trana, P. Childerhouse and E. Deakins. ‘Supply chain information sharing: challenges and risk mitigation strategies’. In: *Journal of Manufacturing Technology Management* 27.8 (2016), pp. 1102–1126 (Cited on pages 40, 71, 167 and 168).

[96] L. Li and H. Zhang. ‘Confidentiality and Information Sharing in Supply Chain Coordination’. In: *Management Science* 54.8 (2008), pp. 1467–1481 (Cited on pages 40 and 171).

[97] Y. Hong, J. Vaidya and S. Wang. ‘A Survey of Privacy-Aware Supply Chain Collaboration: From Theory to Applications’. In: *Journal of Information Systems* 28.1 (2014), pp. 243–268 (Cited on pages 40, 118 and 171).

[98] B. Kuczenski, C. Sahin and A. El Abbadi. ‘Privacy-preserving aggregation in life cycle assessment’. In: *Environment Systems and Decisions* 37.1 (2017), pp. 13–21 (Cited on pages 40, 102, 118 and 171).

[99] F. Kerschbaum. ‘Secure and Sustainable Benchmarking in Clouds - A Multi-Party Cloud Application with an Untrusted Service Provider’. In: *Business & Information Systems Engineering* 3.3 (2011), pp. 135–143 (Cited on pages 40, 171 and 172).

[100] A. Acar, H. Aksu, A. S. Uluagac and M. Conti. ‘A Survey on Homomorphic Encryption Schemes: Theory and Implementation’. In: *ACM Computing Surveys* 51.4 (2018). Article no. 79 (Cited on pages 40, 170 and 171).

[101] A. Shostack. *Threat Modeling: Designing for Security*. John Wiley & Sons, 2014 (Cited on pages 40, 41, 48, 49, 58, 165 and 166).

[102] J. Majava, V. Isoherranen and P. Kess. ‘Business collaboration concepts and implications for companies’. In: *International Journal of Synergy and Research* 2.1-2 (2013), pp. 23–40 (Cited on page 45).

[103] M. Deng, K. Wuyts, R. Scandariato, B. Preneel and W. Joosen. ‘A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements’. In: *Requirements Engineering* 16.1 (2011), pp. 3–32 (Cited on pages 48, 49, 58, 165 and 166).

[104] W. Xiong and R. Lagerström. ‘Threat modeling - A systematic literature review’. In: *Comput. Secur.* 84 (2019), pp. 53–69 (Cited on page 48).

[105] A. Pfitzmann and M. Hansen. *A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management.* 2010 (Cited on pages 58 and 59).

[106] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen and Y. Koucheryavy. ‘Multi-Factor Authentication: A Survey’. In: *Cryptography* 2.1 (2018). Article no. 1 (Cited on page 59).

[107] M. Naor. ‘Deniable Ring Authentication’. In: *Proceedings of Advances in Cryptology - Annual International Cryptology Conference.* Vol. 2442. Lecture Notes in Computer Science. Santa Barbara, California, USA: Springer, 2002, pp. 481–498 (Cited on page 59).

[108] M. Abadi and C. Fournet. ‘Private authentication’. In: *Theoretical Computer Science* 322.3 (2004), pp. 427–476 (Cited on page 59).

[109] B. Carminati. ‘Digital Signatures’. In: *Encyclopedia of Database Systems.* Springer, 2018 (Cited on page 59).

[110] A. Ahadipour and M. Schanzenbach. ‘A Survey on Authorization in Distributed Systems: Information Storage, Data Retrieval and Trust Evaluation’. In: *Proceedings of IEEE Trustcom/BigDataSE/ICESS.* Sydney, Australia: IEEE Computer Society, 2017, pp. 1016–1023 (Cited on page 59).

[111] K. Sethi, A. Majumdar and P. Bera. ‘A novel implementation of parallel homomorphic encryption for secure data storage in cloud’. In: *Proceedings of International Conference on Cyber Security And Protection Of Digital Services.* London, UK: IEEE, 2017, pp. 1–7 (Cited on page 59).

[112] A. J. Menezes, P. C. Van Oorschot and S. A. Vanstone. *Handbook of Applied Cryptography.* First. CRC Press, 1997 (Cited on page 59).

[113] M. Naor and K. Nissim. ‘Communication Complexity and Secure Function Evaluation’. In: *CoRR* cs.CR/0109011 (2001). arXiv: `0109011` (Cited on page 59).

[114] B. Shebaro, O. Oluwatimi and E. Bertino. ‘Context-Based Access Control Systems for Mobile Devices’. In: *IEEE Transactions on Dependable and Secure Computing* 12.2 (2015), pp. 150–163 (Cited on page 59).

[115] C. K. Georgiadis, I. Mavridis, G. Pangalos and R. K. Thomas. 'Flexible team-based access control using contexts'. In: *Proceedings of the 6th Symposium on Access Control Models and Technologies.* Chantilly, Virginia, USA: ACM, 2001, pp. 21–27 (Cited on page 59).

[116] B. Carminati and E. Ferrari. 'Privacy-Aware Collaborative Access Control in Web-Based Social Networks'. In: *Proceedings of IFIP Annual Conference on Data and Applications Security and Privacy.* Vol. 5094. Lecture Notes in Computer Science. London, UK: Springer, 2008, pp. 81–96 (Cited on page 59).

[117] H. Lin, K. Kaur, X. Wang, G. Kaddoum, J. Hu and M. M. Hassan. 'Privacy-Aware Access Control in IoT-enabled Healthcare: A Federated Deep Learning Approach'. In: *IEEE Internet of Things Journal* (2021) (Cited on page 59).

[118] J. Jow, Y. Xiao and W. Han. 'A survey of intrusion detection systems in smart grid'. In: *International Journal of Sensor Networks* 23.3 (2017), pp. 170–186 (Cited on page 59).

[119] B. Sun, L. Osborne, Y. Xiao and S. Guizani. 'Intrusion detection techniques in mobile ad hoc and wireless sensor networks'. In: *IEEE Wireless Communications* 14.5 (2007), pp. 56–63 (Cited on page 59).

[120] X. Liu, X. Yang and Y. Lu. 'To filter or to authorize: network-layer DoS defense against multimillion-node botnets'. In: *Proceedings of the ACM Conference on Data Communication.* Seattle, WA, USA: ACM, 2008, pp. 195–206 (Cited on page 59).

[121] X. Yang, D. Wetherall and T. E. Anderson. 'TVA: a DoS-limiting network architecture'. In: *IEEE/ACM Transactions on Networking* 16.6 (2008), pp. 1267–1280 (Cited on page 59).

[122] M. D. Raimondo and R. Gennaro. 'New Approaches for Deniable Authentication'. In: *Journal of Cryptology* 22.4 (2009), pp. 572–615 (Cited on page 59).

[123] S. Murthy, A. A. Bakar, F. A. Rahim and R. Ramli. 'A Comparative Study of Data Anonymization Techniques'. In: *Proceedings of 5th IEEE International Conference on Big Data Security on Cloud, IEEE International Conference on High Performance and Smart Computing, and IEEE International Conference on Intelligent Data and Security.* Washington, DC, USA: IEEE, 2019, pp. 306–309 (Cited on page 59).

[124] I. S. Moskowitz, R. E. Newman, D. P. Crepeau and A. R. Miller. 'Covert channels and anonymizing networks'. In: *Proceedings of the ACM Workshop on Privacy in the Electronic Society.* Washington, DC, USA: ACM, 2003, pp. 79–88 (Cited on page 59).

[125] R. J. Anderson and F. A. P. Petitcolas. 'On the limits of steganography'. In: *IEEE Journal on Selected Areas in Communications* 16.4 (1998), pp. 474–481 (Cited on page 59).

[126] W. Huang, Y. Liao, S. Zhou and H. Chen. 'An Efficient Deniable Authenticated Encryption Scheme for Privacy Protection'. In: *IEEE Access* 7 (2019), pp. 43453–43461 (Cited on page 59).

[127] M. Waidner and B. Pfitzmann. 'The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability'. In: *Proceedings of Advances in Cryptology — Jean-Jacques Quisquater and Joos Vandewalle.* Vol. 434. Lecture Notes in Computer Science. Springer, 1990, p. 690 (Cited on page 59).

[128] S. Lederer, J. I. Hong, A. K. Dey and J. A. Landay. 'Personal privacy through understanding and action: five pitfalls for designers'. In: *Personal and Ubiquitous Computing* 8.6 (2004), pp. 440–454 (Cited on page 59).

[129] S. Patil and A. Kobsa. 'Privacy Considerations in Awareness Systems: Designing with Privacy in Mind'. In: *Awareness Systems: Advances in Theory, Methodology and Design.* Springer, 2009, pp. 187–206 (Cited on page 59).

[130] L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall and J. Reagle. *The Platform for Privacy Preferences project, W3C P3P specifications.* [Online]. `https://www.w3.org/TR/P3P/`, Accessed: 02.11.2022. 2022 (Cited on pages 59 and 85).

[131] OASIS. *Extensible Access Control Markup Language (XACML).* [Online]. `http://xml.coverpages.org/xacml.html`, Accessed: 02.11.2022. 2022 (Cited on pages 59 and 85).

[132] R. Gennaro, M. O. Rabin and T. Rabin. 'Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography'. In: *Proceedings of the 17th Annual ACM Symposium on Principles of Distributed Computing.* Puerto Vallarta, Mexico: ACM, 1998, pp. 101–111 (Cited on page 81).

[133] A. Khraisat, I. Gondal, P. Vamplew and J. Kamruzzaman. 'Survey of intrusion detection systems: techniques, datasets and challenges'. In: *Cybersecurity* 2.1 (2019), p. 20 (Cited on page 82).

[134] G. Baldini and I. Amerini. 'Online Distributed Denial of Service (DDoS) intrusion detection based on adaptive sliding window and morphological fractal dimension'. In: *Computer Networks* 210 (2022). Article no. 108923 (Cited on page 82).

[135] M. Asim, A. Yautsiukhin, A. D. Brucker, T. Baker, Q. Shi and B. Lempereur. ‘Security policy monitoring of BPMN-based service compositions’. In: *J. Softw. Evol. Process.* 30.9 (2018) (Cited on page 85).

[136] A. D. Brucker, B. Zhou, F. Malmignati, Q. Shi and M. Merabti. ‘Modelling, validating, and ranking of secure service compositions’. In: *Softw. Pract. Exp.* 47.12 (2017), pp. 1923–1943 (Cited on page 85).

[137] P. Ashley, S. Hada, G. Karjoth, C. Powers and M. Schunter. *Enterprise Privacy Authorization Language (EPAL 1.2).* [Online]. `http://www.w3.org/Submission/2003/SUBM-EPAL-20031110/`, Accessed: 02.11.2022. 2022 (Cited on page 85).

[138] E. Barker and Q. Dang. *Recommendation for Key Management, Part 3: Application-Specific Key Management Guidance.* Tech. rep. Gaithersburg, MD, USA: National Institute of Standards and Technology, 2015 (Cited on page 88).

[139] L. Chen, D. Moody, K. Randall, A. Regenscheid and A. Robinson. *Recommendations for Discrete Logarithm-based Cryptography: Elliptic Curve Domain Parameters.* 2023 (Cited on page 88).

[140] A. Aly, K. Cong, D. Cozzo, M. Keller, E. Orsini, D. Rotaru, O. Scherer, P. Scholl, N. Smart, T. Tanguy et al. *SCALE-MAMBA v1.13.* 2021 (Cited on pages 93, 95 and 130).

[141] *SCALE-MAMBA Software.* [Online]. `https://homes.esat.kuleuven.be/~nsmart/SCALE/`, Accessed: 21.01.2023. 2023 (Cited on page 93).

[142] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl and N. P. Smart. ‘Practical Covertly Secure MPC for Dishonest Majority - Or: Breaking the SPDZ Limits’. In: *Proceedings of European Symposium on Research in Computer Security.* Vol. 8134. Lecture Notes in Computer Science. Egham, UK: Springer, 2013, pp. 1–18 (Cited on page 95).

[143] A. Shamir. ‘How to Share a Secret’. In: *Communications of the ACM* 22.11 (1979), pp. 612–613 (Cited on page 95).

[144] U. M. Maurer. ‘Secure multi-party computation made simple’. In: *Discrete Applied Mathematics* 154.2 (2006), pp. 370–381 (Cited on page 95).

[145] M. Keller, D. Rotaru, N. P. Smart and T. Wood. ‘Reducing Communication Channels in MPC’. In: *Proceedings of International Conference on Security and Cryptography for Networks.* Vol. 11035. Lecture Notes in Computer Science. Amalfi, Italy: Springer, 2018, pp. 181–199 (Cited on page 95).

[146] R. Jadoul, N. P. Smart and B. V. Leeuwen. ‘MPC for Q_2 Access Structures over Rings and Fields’. In: *Proceedings of International Conference on Selected Areas in Cryptography*. Vol. 13203. Lecture Notes in Computer Science. Springer, 2021, pp. 131–151 (Cited on pages 95 and 96).

[147] R. Cramer, I. Damgård and U. M. Maurer. ‘General Secure Multi-party Computation from any Linear Secret-Sharing Scheme’. In: *Proceedings of Advances in Cryptology - International Conference on the Theory and Applications of Cryptographic Techniques*. Vol. 1807. Lecture Notes in Computer Science. Bruges, Belgium: Springer, 2000, pp. 316–334 (Cited on page 95).

[148] T. Araki, J. Furukawa, Y. Lindell, A. Nof and K. Ohara. ‘High-Throughput Semi-Honest Secure Three-Party Computation with an Honest Majority’. In: *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*. Vienna, Austria: ACM, 2016, pp. 805–817 (Cited on page 95).

[149] J. Furukawa, Y. Lindell, A. Nof and O. Weinstein. ‘High-Throughput Secure Three-Party Computation for Malicious Adversaries and an Honest Majority’. In: *Proceedings of Advances in Cryptology - Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Vol. 10211. Lecture Notes in Computer Science. Paris, France, 2017, pp. 225–255 (Cited on page 95).

[150] C. Orlandi. *MPC Techniques Series, Part 1: Secret Sharing*. [Online]. `https://medium.com/partisia-blockchain/mpc-techniques-series-part-1-secret-sharing-d8f98324674a`, Accessed: 10.11.2022. 2022 (Cited on page 95).

[151] M. Abdel-Basset, G. Manogaran and M. Mohamed. ‘Internet of Things (IoT) and its impact on supply chain: A framework for building smart, secure and efficient systems’. In: *Future Generation Computer Systems* 86 (2018), pp. 614–628 (Cited on page 102).

[152] T. Norgate, S. Jahanshahi and W. Rankin. ‘Assessing the environmental impact of metal production processes’. In: *Journal of Cleaner Production* 15.8 (2007), pp. 838–848. ISSN: 0959-6526 (Cited on page 102).

[153] *Life Cycle Inventory Data and environmental Metrics for the Primary Aluminium Industry*. `https://web.archive.org/web/20210923150124/https://international-aluminium.org/resource/life-cycle-inventory-data-and-environmental-metrics/`. Sept. 2015 (Cited on pages 102, 103 and 112).

[154] *Life Cycle Inventory Data and environmental Metrics for the Primary Aluminium Industry*. `https://web.archive.org/web/20210928130650/https://international-aluminium.org/wp-content/uploads/2021/03/2015LI1.zip`. Sept. 2015 (Cited on pages 102, 103, 108 and 112).

[155] R. Heijungs and S. Suh. *The Computational Structure of Life Cycle Assessment.* Springer Dordrecht, 2002 (Cited on pages 107 and 108).

[156] M. Hauschild, R. K. Rosenbaum and S. Irving Olsen. *Life Cycle Assessment - Theory and Practice.* Firts. Springer Cham, 2018 (Cited on page 108).

[157] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. *Introduction to algorithms.* Fourth. The MIT Press, 2022 (Cited on pages 131 and 133).

[158] L. Dickmanns and M. von Maltitz. 'Performance of Secure Multiparty Computation'. In: *Proceedings of the Seminar Innovative Internet Technologies and Mobile Communications (IITM).* Ed. by B. J. Georg Carle Stephan Gunther. Chair of Network Architectures and Services, 2019 (Cited on page 162).

[159] Y. Chen, D. Hu and G. Zhang. 'Data Mining and Critical Success Factors in Data Mining Projects'. In: *Proceedings of Knowledge Enterprise: Intelligent Strategies in Product Design, Manufacturing, and Management.* Vol. 207. IFIP. Springer, 2006, pp. 281–287 (Cited on page 164).

[160] F. Gogoulos, A. Antonakopoulou, G. V. Lioudakis, A. S. Mousas, D. I. Kaklamani and I. S. Venieris. 'On the design of a privacy aware authorization engine for collaborative environments'. In: *Electronic Markets* 24.2 (2014), pp. 101–112 (Cited on pages 165 and 167).

[161] P. D. Ray, R. Harnoor and M. Hentea. 'Smart power grid security: A unified risk management approach'. In: *Proceedings of the 44th Annual 2010 IEEE International Carnahan Conference on Security Technology.* IEEE. 2010, pp. 276–285 (Cited on page 165).

[162] C. Udokwu and A. Norta. 'Deriving and formalizing requirements of decentralized applications for inter-organizational collaborations on blockchain'. In: *Arabian Journal for Science and Engineering* 46.9 (2021), pp. 8397–8414 (Cited on page 165).

[163] B. Schneier. 'Attack trees'. In: *Dr. Dobb's journal* 24.12 (1999), pp. 21–29 (Cited on page 165).

[164] R. Clarke. 'Privacy impact assessment: Its origins and development'. In: *Computer Law & Security Review* 25.2 (2009), pp. 123–135 (Cited on page 165).

[165] D. J. Solove. 'A taxonomy of privacy'. In: *University of Pennsylvania Law Review* 154.3 (2005), pp. 477–564 (Cited on pages 165 and 166).

[166] A. S. Sodiya, S. A. Onashoga and B. Oladunjoye. 'Threat modeling using fuzzy logic paradigm'. In: *Informing Science: International Journal of an Emerging Transdiscipline* 4.1 (2007), pp. 53–61 (Cited on page 165).

[167] A. Shostack. 'Experiences Threat Modeling at Microsoft'. In: *Proceedings of Workshop on Modeling Security.* Vol. 413. CEUR Workshop Proceedings. Lancaster, UK: CEUR-WS.org, 2008 (Cited on page 165).

[168] A. Shostack. *Security Briefs - Getting Started With The SDL Threat Modeling Tool.* [Online]. `https://docs.microsoft.com/en-us/archive/msdn-magazine/2009/january/security-briefs-getting-started-with-the-sdl-threat-modeling-tool`, Accessed: 25.09.2022. 2022 (Cited on page 165).

[169] Y. Chen, B. W. Boehm and L. Sheppard. 'Value Driven Security Threat Modeling Based on Attack Path Analysis'. In: *Proceedings of the 40th Annual Hawaii International Conference on System Sciences.* Big Island, HI, USA: IEEE Computer Society, 2007, 280a–280a (Cited on page 166).

[170] V. Saini, Q. Duan and V. Paruchuri. 'Threat modeling using attack trees'. In: *Journal of Computing Sciences in Colleges* 23.4 (2008), pp. 124–131 (Cited on page 166).

[171] F. Den Braber, I. Hogganvik, M. S. Lund, K. Stølen and F. Vraalsen. 'Model-based security analysis in seven steps—a guided tour to the CORAS method'. In: *BT Technology Journal* 25.1 (2007), pp. 101–117 (Cited on pages 166 and 168).

[172] I. Band, W. Engelsman, C Feltus, S. G. Paredes and D. Diligens. 'Modeling enterprise risk management and security with the archimate'. In: *The Open Group, White Paper* (2015) (Cited on pages 166 and 168).

[173] A. de Ruijter and F. Guldenmund. 'The bowtie method: A review'. In: *Safety Science* 88 (2016), pp. 211–218 (Cited on pages 166 and 168).

[174] A. Cooper, H. Tschofenig, B. Aboba, J. Peterson, J. B. Morris, M. Hansen and R. Smith. 'Privacy Considerations for Internet Protocols'. In: *RFC* 6973 (2013), pp. 1–36 (Cited on page 166).

[175] D. Wright. 'The state of the art in privacy impact assessment'. In: *Computer Law & Security Review* 28.1 (2012), pp. 54–61 (Cited on page 166).

[176] D. J. Solove. *Understanding privacy.* Tech. rep. The George Washington University Law School, 2008 (Cited on page 166).

[177] S. Pandey, R. K. Singh, A. Gunasekaran and A. Kaushik. 'Cyber security risks in globalized supply chains: conceptual framework'. In: *Journal of Global Operations and Strategic Sourcing* 13.1 (2020), pp. 103–128 (Cited on page 167).

[178] Y. Tu, W. Zhou and S. Piramuthu. 'Critical risk considerations in auto-ID security: Barcode vs. RFID'. In: *Decision Support Systems* 142 (2021). Article no. 113471 (Cited on page 167).

[179] A. D. Brucker, I. Hang, G. Lückemeyer and R. Ruparel. 'SecureBPMN: modeling and enforcing access control requirements in business processes'. In: *Proceedings of the 17th ACM symposium on Access Control Models and Technologies.* ACM, 2012, pp. 123–126 (Cited on page 167).

[180] H. Irshad, B. Shafiq, J. Vaidya, M. A. Bashir, S. Shamail and N. R. Adam. 'Preserving Privacy in Collaborative Business Process Composition'. In: *Proceedings of the 12th International Joint Conference on e-Business and Telecommunications*. SciTePress, 2015, pp. 112–123 (Cited on page 167).

[181] S. Belluccini, R. D. Nicola, M. Dumas, P. Pullonen, B. Re and F. Tiezzi. 'Verification of Privacy-Enhanced Collaborations'. In: *Proceedings of the 8th International Conference on Formal Methods in Software Engineering*. ACM, 2020, pp. 141–152 (Cited on page 167).

[182] A. D. Brucker and I. Hang. 'Secure and Compliant Implementation of Business Process-Driven Systems'. In: *Proceedings of International Conference on Business Process Management*. Berlin, Heidelberg: Springer, 2013, pp. 662–674 (Cited on page 167).

[183] M. Salnitri, F. Dalpiaz and P. Giorgini. 'Designing secure business processes with SecBPMN'. In: *Software and Systems Modeling* 16.3 (2017), pp. 737–757 (Cited on page 167).

[184] M. Asim, A. Yautsiukhin, A. D. Brucker, T. Baker, Q. Shi and B. Lempereur. 'Security Policy Monitoring of BPMN-based Service Compositions'. In: *Journal of Software: Evolution and Process* 30.9 (2018). Article no. e1944 (Cited on page 167).

[185] L. Hotz, S. von Riegen, A. Pokahr, L. Braubach and T. Schwinghammer. 'Monitoring BPMN-Processes with Rules in a Distributed Environment'. In: *Proceedings of RuleML2012@ECAI Challenge, at the 6th International Symposium on Rules*. 874. CEUR-WS.org, 2012 (Cited on page 167).

[186] W. Labda, N. Mehandjiev and P. Sampaio. 'Modeling of privacy-aware business processes in BPMN to protect personal data'. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. ACM, 2014, pp. 1399–1405 (Cited on page 167).

[187] A. Toots, R. Tuuling, M. Yerokhin, M. Dumas, L. García-Bañuelos, P. Laud, R. Matulevicius, A. Pankova, M. Pettai, P. Pullonen et al. 'Business Process Privacy Analysis in Pleak'. In: *Proceedings of International Conference on Fundamental Approaches to Software Engineering*. Vol. 11424. Springer, 2019, pp. 306–312 (Cited on page 167).

[188] R. Accorsi and A. Lehmann. 'Automatic Information Flow Analysis of Business Process Models'. In: *Proceedings of International Conference on Business Process Management*. Vol. 7481. Springer, 2012, pp. 172–187 (Cited on page 167).

[189] A. Poller, S. Türpe and K. Kinder-Kurlanda. ‘An asset to security modeling? Analyzing stakeholder collaborations instead of threats to assets’. In: *Proceedings of New Security Paradigms Workshop.* 2014, pp. 69–82 (Cited on page 168).

[190] C. Liu, Q. Li and X. Zhao. ‘Challenges and opportunities in collaborative business process management: Overview of recent advances and introduction to the special issue’. In: *Information Systems Frontiers* 11.3 (2009), pp. 201–209 (Cited on pages 168 and 169).

[191] J. Pennekamp, R. Glebke, M. Henze, T. Meisen, C. Quix, R. Hai, L. C. Gleim, P. Niemietz, M. Rudack, S. Knape et al. ‘Towards an Infrastructure Enabling the Internet of Production’. In: *Proceedings of IEEE International Conference on Industrial Cyber Physical Systems.* Taipei, Taiwan: IEEE, 2019, pp. 31–37 (Cited on page 168).

[192] W. M. P. van der Aalst and M. Weske. ‘The P2P Approach to Interorganizational Workflows’. In: *Proceedings of International Conference on Advanced Information Systems Engineering.* Lecture Notes in Computer Science. Interlaken, Switzerland: Springer, 2001, pp. 140–156 (Cited on page 169).

[193] Z. Shan, D. K. W. Chiu and Q. Li. ‘Systematic Interaction Management in a Workflow View Based Business-to-Business Process Engine’. In: *Proceedings of the 38th Annual Hawaii International Conference on System Sciences.* Big Island, HI, USA: IEEE Computer Society, 2005, 162b–162b (Cited on page 169).

[194] X. Zhao, C. Liu and Y. Yang. ‘An Organisational Perspective on Collaborative Business Processes’. In: *Proceedings of International Conference on Business Process Management.* Vol. 3649. Nancy, France, 2005, pp. 17–31 (Cited on page 169).

[195] I. Chebbi, S. Dustdar and S. Tata. ‘The view-based approach to dynamic inter-organizational workflow cooperation’. In: *Data & Knowledge Engineering* 56.2 (2006), pp. 139–173 (Cited on page 169).

[196] O. Perrin and C. Godart. ‘A model to support collaborative work in virtual enterprises’. In: *Data & Knowledge Engineering* 50.1 (2004), pp. 63–86 (Cited on page 169).

[197] C. Dwork. ‘Differential Privacy’. In: *Proceedings of Automata, Languages and Programming - International Colloquium on Automata, Languages, and Programming.* Vol. 4052. Lecture Notes in Computer Science. Venice, Italy: Springer, 2006, pp. 1–12 (Cited on page 169).

[198] C. Dwork and A. Roth. ‘The Algorithmic Foundations of Differential Privacy’. In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407 (Cited on pages 169 and 170).

[199] M. A. Husnoo, A. Anwar, R. K. Chakrabortty, R. Doss and M. Ryan. ‘Differential Privacy for IoT-Enabled Critical Infrastructure: A Comprehensive Survey’. In: *IEEE Access* 9 (2021), pp. 153276–153304 (Cited on pages 169 and 170).

[200] Q. Geng and P. Viswanath. ‘The Optimal Noise-Adding Mechanism in Differential Privacy’. In: *IEEE Transactions on Information Theory* 62.2 (2016), pp. 925–951 (Cited on page 169).

[201] A. Machanavajjhala, X. He and M. Hay. ‘Differential Privacy in the Wild: A Tutorial on Current Practices & Open Challenges’. In: *Proceedings of ACM International Conference on Management of Data*. Chicago, IL, USA: ACM, 2017, pp. 1727–1730 (Cited on page 169).

[202] T. Zhu, G. Li, W. Zhou and P. S. Yu. ‘Preliminary of differential privacy’. In: *Differential Privacy and Applications*. Springer, 2017, pp. 7–16 (Cited on page 170).

[203] J. Domingo-Ferrer, D. Sánchez and A. Blanco-Justicia. ‘The limits of differential privacy (and its misuse in data release and machine learning)’. In: *Communications of the ACM* 64.7 (2021), pp. 33–35 (Cited on page 170).

[204] P. Jain, M. Gyanchandani and N. Khare. ‘Differential privacy: its technological prescriptive using big data’. In: *Journal of Big Data* 5.1 (2018), p. 15 (Cited on page 170).

[205] A. H. Mondal, M. Ranjan and M. Saikia. ‘A brief overview of homomorphic cryptosystem and their applications’. In: *International Journal of Computers and Applications* 975 (2015), p. 8887 (Cited on page 170).

[206] A. S. Sheela and R. G. Franklin. ‘E-Voting System Using Homomorphic Encryption Technique’. In: *Proceedings of Journal of Physics: Conference Series*. Vol. 1770. 1. Article no. 012011. IOP Publishing. 2021 (Cited on page 170).

[207] R. Suwandi, S. M. Nasution and F. Azmi. ‘Secure E-voting system by utilizing homomorphic properties of the encryption algorithm’. In: *Telkomnika (Telecommunication Computing Electronics and Control)* 16.2 (2018), pp. 862–867 (Cited on page 170).

[208] Y. Lu and M. Zhu. ‘Privacy preserving distributed optimization using homomorphic encryption’. In: *Automatica* 96 (2018), pp. 314–325 (Cited on page 170).

[209] D. Wang, B. Guo, Y. Shen, S.-J. Cheng and Y.-H. Lin. ‘A faster fully homomorphic encryption scheme in big data’. In: *Proceedings of IEEE 2nd International Conference on Big Data Analysis*. 2017, pp. 345–349 (Cited on page 170).

[210] N. G. Tsoutsos and M. Maniatakos. ‘Efficient Detection for Malicious and Random Errors in Additive Encrypted Computation’. In: *IEEE Transactions on Computers* 67.1 (2018), pp. 16–31 (Cited on page 170).

[211] N. Emmanuel, A. Khan, M. Alam, T. Khan and M. K. Khan. ‘Structures and data preserving homomorphic signatures’. In: *Journal of Network and Computer Applications* 102 (2018), pp. 58–70 (Cited on page 170).

[212] T. Dugan and X. Zou. ‘Privacy-preserving evaluation techniques and their application in genetic tests’. In: *Smart Health* 1-2 (2017). Connected Health: Applications, Systems and Engineering Technologies 2016, pp. 2–17 (Cited on page 170).

[213] C. Stergiou, K. E. Psannis, B. B. Gupta and Y. Ishibashi. ‘Security, privacy & efficiency of sustainable Cloud Computing for Big Data & IoT’. In: *Sustainable Computing: Informatics and Systems* 19 (2018), pp. 174–184 (Cited on page 170).

[214] X. Jie and R. J. Jing. ‘On-Line Decrypting: A Homomorphic Realization for Network Coding’. In: *Proceedings of Vehicle, Mechatronics and Information Technologies II*. Vol. 543. Applied Mechanics and Materials. Trans Tech Publications Ltd, 2014, pp. 2728–2732 (Cited on page 170).

[215] V. J. Winkler. *Securing the Cloud: Cloud computer Security techniques and tactics*. First. Elsevier, 2011 (Cited on page 170).

[216] M. Christodorescu. ‘Private use of untrusted web servers via opportunistic encryption’. In: *Proceedings of Web 2.0 Security and Privacy Workshop*. 2008 (Cited on page 170).

[217] M. A. Shibli, I. Yousaf and S. Muftic. ‘MagicNET: Security System for Protection of Mobile Agents’. In: *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*. Perth, Australia: IEEE Computer Society, 2010, pp. 1233–1240 (Cited on page 170).

[218] M. Alloghani, M. M. Alani, D. Al-Jumeily, T. Baker, J. Mustafina, A. Hussain and A. J. Aljaaf. ‘A systematic review on the status and progress of homomorphic encryption technologies’. In: *Journal of Information Security and Applications* 48 (2019). Article no. 102362 (Cited on page 170).

[219] F. Armknecht, C. Boyd, C. Carr, K. Gjøsteen, A. Jäschke, C. A. Reuter and M. Strand. ‘A Guide to Fully Homomorphic Encryption’. In: *IACR Cryptology ePrint Archive* (2015), p. 1192 (Cited on page 171).

[220] B. Schwarzbach, M. Glöckner, S. Makarov, B. Franczyk and A. Ludwig. ‘Privacy Preserving BPMS for Collaborative BPaaS’. In: *Proceedings of Federated Conference on Computer Science and Information Systems*. Annals of Computer Science and Information Systems. 2017, pp. 925–934 (Cited on page 171).

[221] B. Carminati, E. Ferrari and C. Rondanini. ‘Blockchain as a Platform for Secure Inter-Organizational Business Processes’. In: *Proceedings of the 4th IEEE International Conference on Collaboration and Internet Computing*. Philadelphia, PA, USA: IEEE Computer Society, 2018, pp. 122–129 (Cited on page 172).

[222] A. E. Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou. ‘Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts’. In: *Proceedings of IEEE Symposium on Security and Privacy*. San Jose, CA, USA: IEEE Computer Society, 2016, pp. 839–858 (Cited on page 172).

[223] P. McCorry, S. F. Shahandashti and F. Hao. ‘A Smart Contract for Boardroom Voting with Maximum Voter Privacy’. In: *Proceedings of International Conference on Financial Cryptography and Data Security*. Vol. 10322. Lecture Notes in Computer Science. Sliema, Malta: Springer, 2017, pp. 357–375 (Cited on page 172).

[224] G. Zyskind, O. Nathan and A. Pentland. ‘Enigma: Decentralized Computation Platform with Guaranteed Privacy’. In: *CoRR* abs/1506.03471 (2015). arXiv: `1506.03471` (Cited on page 172).

[225] D. C. Sánchez. ‘Raziel: Private and Verifiable Smart Contracts on Blockchains’. In: *CoRR* abs/1807.09484 (2018). arXiv: `1807.09484` (Cited on page 172).

[226] B. Carminati, C. Rondanini and E. Ferrari. ‘Confidential Business Process Execution on Blockchain’. In: *Proceedings of IEEE International Conference on Web Services*. San Francisco, CA, USA: IEEE, 2018, pp. 58–65 (Cited on page 172).

[227] T. Hiessl, D. Schall, J. Kemnitz and S. Schulte. ‘Industrial Federated Learning - Requirements and System Design’. In: *Proceedings of International Conference on Practical Applications of Agents and Multi-Agent Systems*. Vol. 1233. Communications in Computer and Information Science. L’Aquila, Italy: Springer, 2020, pp. 42–53 (Cited on page 173).

[228] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings et al. ‘Advances and Open Problems in Federated Learning’. In: *Foundations and Trends in Machine Learning* 14.1-2 (2021), pp. 1–210 (Cited on page 173).

[229] T. Hiessl, S. R. Lakani, J. Kemnitz, D. Schall and S. Schulte. ‘Cohort-based federated learning services for industrial collaboration on the edge’. In: *Journal of Parallel and Distributed Computing* 167 (2022), pp. 64–76 (Cited on page 173).

[230] V. Mugunthan, A. Polychroniadou, D. Byrd and T. H. Balch. ‘Smpai: Secure multi-party computation for federated learning’. In: *Proceedings of the 33rd Conference on Neural Information Processing Systems*. Vancouver, Canada, 2019 (Cited on page 173).

[231] S. Sharma, C. Xing, Y. Liu and Y. Kang. ‘Secure and Efficient Federated Transfer Learning’. In: *Proceedings of IEEE International Conference on Big Data*. Los Angeles, CA, USA: IEEE, 2019, pp. 2569–2576 (Cited on page 173).

[232] H. Zhu, Z. Li, M. Cheah and R. S. M. Goh. ‘Privacy-preserving Weighted Federated Learning within Oracle-Aided MPC Framework’. In: *CoRR* abs/2003.07630 (2020). arXiv: 2003.07630 (Cited on page 173).

[233] J. So, B. Guler and S. Avestimehr. ‘A Scalable Approach for Privacy-Preserving Collaborative Machine Learning’. In: *Proceedings of Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 8054–8066 (Cited on page 173).

[234] X. Yin, Y. Zhu and J. Hu. ‘A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions’. In: *ACM Computing Surveys* 54.6 (2021). Article no. 131, pp. 1–36 (Cited on page 173).

[235] X. Zheng and Z. Cai. ‘Privacy-Preserved Data Sharing Towards Multiple Parties in Industrial IoTs’. In: *IEEE Journal on Selected Areas in Communications* 38.5 (2020), pp. 968–979 (Cited on page 173).

[236] L. Lyu, K. Nandakumar, B. I. P. Rubinstein, J. Jin, J. Bedo and M. Palaniswami. ‘PPFA: Privacy Preserving Fog-Enabled Aggregation in Smart Grid’. In: *IEEE Transactions on Industrial Informatics* 14.8 (2018), pp. 3733–3744 (Cited on page 173).

[237] R. Lu, K. Heung, A. H. Lashkari and A. A. Ghorbani. ‘A Lightweight Privacy-Preserving Data Aggregation Scheme for Fog Computing-Enhanced IoT’. In: *IEEE Access* 5 (2017), pp. 3302–3312 (Cited on page 173).

[238] S. Li, S. Zhao, G. Min, L. Qi and G. Liu. ‘Lightweight Privacy-Preserving Scheme Using Homomorphic Encryption in Industrial Internet of Things’. In: *IEEE Internet of Things Journal* 9.16 (2022), pp. 14542–14550 (Cited on page 173).

[239] W. Ren, X. Tong, J. Du, N. Wang, S. Li, G. Min, Z. Zhao and A. K. Bashir. ‘Privacy-preserving using homomorphic encryption in Mobile IoT systems’. In: *Computer Communications* 165 (2021), pp. 105–111 (Cited on page 173).

[240] J. Lopez-Fenner, S. Sepulveda, L. F. Bittencourt, F. M. Costa and N. Georgantas. 'Privacy Preserving Multi Party Computation for Data-Analytics in the IoT-Fog-Cloud Ecosystem'. In: *Proceedings of IV International Congress of Computer Sciences and Information Systems*. CICCSI 2020 Proceedings. Mendoza / Virtual, Argentina, 2020 (Cited on page 174).

[241] D. Demmler, T. Schneider and M. Zohner. 'ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation'. In: *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*. The Internet Society, 2015 (Cited on page 174).

[242] X. Wang, S. Ranellucci and J. Katz. *Global-Scale Secure Multiparty Computation*. Cryptology ePrint Archive, Paper 2017/189. `https://eprint.iacr.org/2017/189`. 2017 (Cited on page 174).

[243] M. Keller. 'MP-SPDZ: A Versatile Framework for Multi-Party Computation'. In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. CCS '20. New York, NY, USA: Association for Computing Machinery, 2020, 1575–1590. ISBN: 9781450370899 (Cited on pages 174, 176 and 180).

[244] B. Schoenmakers. 'MPyC—Python package for secure multiparty computation'. In: *Workshop on the Theory and Practice of MPC. https://github.com/lschoe/mpyc*. 2018 (Cited on page 174).

[245] S. Zahur and D. Evans. *Obliv-C: A Language for Extensible Data-Oblivious Computation*. Cryptology ePrint Archive, Paper 2015/1153. `https://eprint.iacr.org/2015/1153`. 2015 (Cited on page 174).

[246] C. Liu, X. S. Wang, K. Nayak, Y. Huang and E. Shi. 'ObliVM: A Programming Framework for Secure Computation'. In: *2015 IEEE Symposium on Security and Privacy*. 2015, pp. 359–376 (Cited on page 174).

[247] Y. Zhang, A. Steele and M. Blanton. 'PICCO: A General-Purpose Compiler for Private Distributed Computation'. In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. CCS '13. New York, NY, USA: Association for Computing Machinery, 2013, 813–826. ISBN: 9781450324779 (Cited on pages 174 and 180).

[248] A. Aly, K. Cong, D. Cozzo, M. Keller, E. Orsini, D. Rotaru, O. Scherer, P. Scholl, N. Smart, T. Tanguy et al. *SCALE-MAMBA v1.14*. 2021 (Cited on pages 174 and 176).

[249] D. Bogdanov, S. Laur and J. Willemson. ‘Sharemind: A Framework for Fast Privacy-Preserving Computations’. In: *Computer Security - ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings*. Ed. by S. Jajodia and J. López. Vol. 5283. Lecture Notes in Computer Science. Springer, 2008, pp. 192–206 (Cited on page 174).

[250] M. Hastings, B. Hemenway, D. Noble and S. Zdancewic. ‘SoK: General Purpose Compilers for Secure Multi-Party Computation’. In: *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 2019, pp. 1220–1237 (Cited on pages 174, 175 and 176).

[251] *Multi-Protocol SPDZ*. [Online]. `https://github.com/data61/MP-SPDZ` Accessed: 20.07.2023. 2023 (Cited on page 176).

[252] C. Bonte, N. P. Smart and T. Tanguy. ‘Thresholdizing HashEdDSA: MPC to the Rescue’. In: *Int. J. Inf. Sec.* 20.6 (2021), pp. 879–894 (Cited on page 176).

[253] G. Büyüközkan and J. Arsenyan. ‘Collaborative product development: a literature overview’. In: *Production Planning & Control* 23.1 (2012), pp. 47–66 (Cited on page 178).

[254] L. Zhang, J. Xu, P. Vijayakumar, P. K. Sharma and U. Ghosh. ‘Homomorphic Encryption-based Privacy-preserving Federated Learning in IoT-enabled Healthcare System’. In: *IEEE Transactions on Network Science and Engineering* (2022), pp. 1–17 (Cited on page 178).

[255] H. M. Khan, A. Khan, F. Jabeen, A. Anjum and G. Jeon. ‘Fog-enabled secure multiparty computation based aggregation scheme in smart grid’. In: *Computers & Electrical Engineering* 94 (2021), p. 107358. ISSN: 0045-7906 (Cited on page 178).

[256] T. K. Frederiksen, B. Pinkas and A. Yanai. ‘Committed MPC’. In: *Public-Key Cryptography – PKC 2018*. Ed. by M. Abdalla and R. Dahab. Springer International Publishing, 2018, pp. 587–619. ISBN: 978-3-319-76578-5 (Cited on page 178).

[257] T. P. Pedersen. ‘Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing’. In: *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*. CRYPTO ’91. Berlin, Heidelberg: Springer-Verlag, 1991, 129–140. ISBN: 3540551883 (Cited on page 178).

[258] Y. Lindell and B. Pinkas. *An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries*. Cryptology ePrint Archive, Paper 2008/049. `https://eprint.iacr.org/2008/049`. 2008 (Cited on page 178).

[259] I. Damgård and M. Jurik. 'A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System'. In: *Public Key Cryptography.* Ed. by K. Kim. Springer Berlin Heidelberg, 2001, pp. 119–136. ISBN: 978-3-540-44586-9 (Cited on page 178).

[260] S. Mödersheim and L. Viganò. 'Alpha-Beta Privacy'. In: *ACM Trans. Priv. Secur.* 22.1 (2019). ISSN: 2471-2566 (Cited on page 179).

[261] R. Marinho and R. Holanda. 'Automated Emerging Cyber Threat Identification and Profiling Based on Natural Language Processing'. In: *IEEE Access* 11 (2023), pp. 58915–58936 (Cited on page 179).

[262] A. Parisi. *Hands-On Artificial Intelligence for Cybersecurity: Implement smart AI systems for preventing cyber attacks and detecting threats and network anomalies.* First. Packt Publishing Ltd, 2019 (Cited on page 179).

[263] T. C. Truong, Q. B. Diep and I. Zelinka. 'Artificial Intelligence in the Cyber Domain: Offense and Defense'. In: *Symmetry* 12.3 (2020). ISSN: 2073-8994 (Cited on page 179).

[264] A. Schaad and D. Binder. 'ML-Supported Identification and Prioritization of Threats in the OVVL Threat Modelling Tool'. In: *Data and Applications Security and Privacy XXXIV - 34th Annual IFIP WG 11.3 Conference, DBSec 2020, Regensburg, Germany, June 25-26, 2020, Proceedings.* Ed. by A. Singhal and J. Vaidya. Vol. 12122. Lecture Notes in Computer Science. Springer, 2020, pp. 274–285 (Cited on page 179).

[265] S. M. Mohammad and L. Surya. 'Security Automation in Information Technology'. In: *International journal of creative research thoughts (IJCRT)–Volume* 6 (2 2018) (Cited on page 179).

[266] M. Ebrahimi, J. F. N. Jr. and H. Chen. 'Semi-Supervised Cyber Threat Identification in Dark Net Markets: A Transductive and Deep Learning Approach'. In: *J. Manag. Inf. Syst.* 37.3 (2020), pp. 694–722 (Cited on page 179).

[267] O. Foundation. *OWASP Threat Dragon.* [Online]. `https://owasp.org/www-project-threat-dragon/`, Accessed: 23.01.2023. 2023 (Cited on page 179).

[268] I. Kunz and A. Binder. 'Application-Oriented Selection Of Privacy Enhancing Technologies'. In: *Privacy Technologies and Policy.* Warsaw, Poland: Springer-Verlag, 2022, pp. 75–87. ISBN: 978-3-031-07314-4 (Cited on page 180).

[269] C. Namugenyi, S. L. Nimmagadda and T. Reiners. 'Design of a SWOT Analysis Model and its Evaluation in Diverse Digital Business Ecosystem Contexts'. In: *Procedia Computer Science* 159 (2019). Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 23rd International Conference KES2019, pp. 1145–1154. ISSN: 1877-0509 (Cited on page 180).

[270] M. A. Benzaghta, A. Elwalda, M. M. Mousa, I. Erkan and M. Rahman. ‘SWOT analysis applications: An integrative literature review’. In: *Journal of Global Business Insights* 6.1 (2021), pp. 55–73 (Cited on page 180).

[271] A. Galazkiewicz and A. Wójtowicz. ‘Multiparty Computation in Practice: Increasing Security of Documents in Enterprise Content Management Systems’. In: *ICSEB 2020: The 4th International Conference on Software and e-Business, Osaka, Japan, December, 2020.* ACM, 2020, pp. 1–6 (Cited on page 180).

[272] S. U. Hussain, B. Li, F. Koushanfar and R. Cammarota. ‘TinyGarble2: Smart, Efficient, and Scalable Yao’s Garble Circuit’. In: *PPMLP’20: Proceedings of the 2020 Workshop on Privacy-Preserving Machine Learning in Practice, Virtual Event, USA, November, 2020.* Ed. by B. Zhang, R. A. Popa, M. Zaharia, G. Gu and S. Ji. ACM, 2020, pp. 65–67 (Cited on page 180).

[273] X. Zhang, C. Shan and Y. Zou. ‘Multi-party Secure Comparison of Strings Based on Outsourced Computation’. In: *Machine Learning for Cyber Security - 4th International Conference, ML4CS 2022, Guangzhou, China, December 2-4, 2022, Proceedings, Part II.* Ed. by Y. Xu, H. Yan, H. Teng, J. Cai and J. Li. Vol. 13656. Lecture Notes in Computer Science. Springer, 2022, pp. 15–30 (Cited on page 180).