Delft University of Technology, Bachelor Seminar of Computer Science and Engineering

A Survey on the Privacy-Preserving Applications of Secure Transformation in Collaborative Supply Chains

Author:Andrei-Alexandru Stefan , Supervisor:Tianyu Li , Responsible Professor:Zekeriya Erkin
Cyber Security Group
Department of Intelligent Systems
Delft University of Technology
{Tianyu.Li, Z.Erkin}@tudelft.nl

June 27, 2021

Abstract

Collaboration is a key technique in modern supply chains, both for building trust with other companies, but also for reducing costs or maximizing profits. It is an approach which provides all involved parties with benefits that they could not possibly achieve on their own. Collaboration, however, requires abundant information, including proprietary information which the owners might not want to disclose publicly. This leads to the main privacy concern, namely ensuring the privacy of proprietary information, since access to this information can mean a competitive advantage in the market. Several techniques which enable collaboration while preserving privacy have been developed over time, including secure transformation, which is a non-cryptographic approach. The main focus of this paper is studying this technique and its recent developments, along with the feasibility of using it to preserve privacy in supply chains, through a literature review. Secure transformation is still somewhat in its infancy, with much theoretical research being conducted, yet the technique still not being employed in practice. Therefore, reviewing the research done is the most suitable approach to answering the question of how secure transformation applications can preserve privacy in collaborative supply chains. The main result of this research is that secure transformation is a double-edged sword which promises effective computation for certain collaborative problems, with the downside of having weaker security guarantees compared to cryptographic approaches.

1 Introduction

A supply chain is a network composed of suppliers, manufacturers, retailers, and customers, having as a main purpose the production and sale of goods. Supply chains range from simple structures made up of one of each member mentioned before, to complex structures with multiple instances of each. In the latter case, collaboration becomes a prevalent approach to reducing costs (for example delivery costs, by sharing supply trucks or shipment containers) while maximizing profits. Such supply chains are called ”collaborative supply chains”, and a common concern is the privacy of each collaborator’s proprietary information. In order to collaborate properly, in theory, all of the participants should have access to each other’s information, in order to solve the problem of cost minimization. This, however, is normally infeasible, as gaining access to private information usually means gaining a competitive advantage. Therefore, the main goal of this area of research is finding techniques which facilitate collaboration, while preserving the privacy of the collaborators.

Collaboration techniques in supply chains have been gaining increasing popularity, as teamwork becomes increasingly essential for both efficiency [1] and sustainability [2]. A recent survey on collaborative supply chain techniques which preserve privacy is [1]. The research question of this paper is ”How is privacy preserved through applications of privacy-preserving techniques for supply chain collaboration among multiple parties?”. The survey presents the concepts of horizontal (collaborators have the same role; for example, both are suppliers) and vertical (collaborators have distinct roles; for example, one is the manufacturer and the other is the retailer) collaboration, along with two approaches to preserve privacy: a trusted third party and no trusted third party. For the first case, the collaborators give all of their private information to the trusted third party, whose goal is to find a solution which is optimal for all participants. This approach has many disadvantages, among which are the cost of hiring the third party and the need to fully trust this third party. The alternative is to use techniques which do not require a third party, and these are secure multi-party computation (MPC) and secure transformation. These techniques are used to compute a function which represents the optimal solution (given as input the private information of all parties involved). Both MPC and secure transformation have their advantages and disadvantages, which will be discussed in a later section. Aside from the challenges mentioned in the paper itself, this survey was published more than 7 years ago (in 2014), and it is likely that new applications, which solve the problem of collaborating while preserving privacy more efficiently, have been developed.

Thus, the main research question of this paper is ”How do secure transformation applications preserve privacy in collaborative supply chains?”. The contents of this research include an overview of both historical and recent developments which relate to the technique of secure transformation and its applications, and also recent developments for other privacy-preserving techniques. These recent developments are compared against each other, and their viability for real-life collaborative applications is considered.

2 Techniques for Privacy Preservation in Supply Chain Collaboration

In this section, the definitions and explanations given in [1] are heavily used as a base for describing techniques with and without a trusted third party.

2.1 Trusted Third Party

A trusted third party is an outside party (meaning it is not one of the collaborators) whose goal is to ensure privacy-preserving collaboration in the supply chain. While it is a convenient approach, as the collaborators themselves do not have to do anything besides provide their private information, it also comes with many drawbacks. Among those, the need to fully trust the third party and the high monetary costs are the biggest downsides which make the trusted third party a sub-optimal choice in many situations.

2.2 No Trusted Third Party

When no trusted third party is present, the collaborators have to organize themselves in order to collaborate properly without revealing private information. In order to do this, the optimal solution is (most often, although there are also other approaches) considered to be the output of a function with multiple inputs (the private information of the collaborators; this could be, for example, the rate at which some product is being manufactured). Thus, the collaborators are looking for a solution to a function. The techniques used to accomplish this are secure multi-party computation (MPC) and secure transformation.

Secure Multi-Party Computation (MPC)

MPC is based on interpreting the function as a combinatorial circuit and giving each collaborator random input and output wires of this circuit. This technique is often referred to as ”garbled circuits”, and while it is not the only technique developed, it is one of the earliest ones. A visual representation of a two party garbled circuits computation can be found in Figure 1. ”The celebrated results (Yao 1986 [3]; Goldreich, Micali, and Wigderson 1987 [4]; Ben-Or, Goldwasser, and Wigderson 1998 [5]) in this area show that any function can be securely computed in a distributed manner efficiently (e.g., in polynomial time with respect to the size of the circuit required to compute the function)” [1, p. 249]. The main advantage of this approach is the provable security, due to it being a cryptograhic approach, and the possibility of computing any function. However, this comes with the disadvantage of being inefficient in most cases, and thus too slow to be used in supply chains [6].

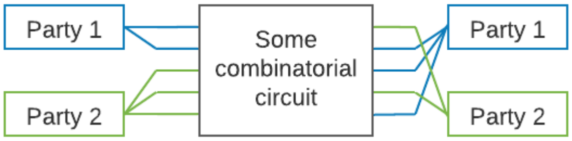

Figure 1: Visualization of garbled circuits (two party case)

Secure Transformation (or Data Obfuscation)

Secure transformation means transforming (or obfuscating) the input into a randomized format such that, when the function is computed, the result does not reveal anything about the input to the other parties. After the transformation, the input is shared among all collaborators and the function also needs to be transformed, to accommodate the transformed inputs. At this point, every party involved has all the information needed to obtain the output of the function. Each collaborator can then find their solution to the original problem, and, more importantly, they **cannot** find the solutions of others. Generally, the problems solved are linear programming problems, meaning that the function is linear. Figure 2 provides the flow of the general secure transformation approach for the two party case. While the efficiency of this approach is promising, secure transformation is, however, not without fault, as ”the secure transformation-based approaches cannot always be considered as ’Provably Secure’ (uncertain privacy risk may still exist)” [1, p. 249], which stems from the non-cryptographic origin of the technique itself.

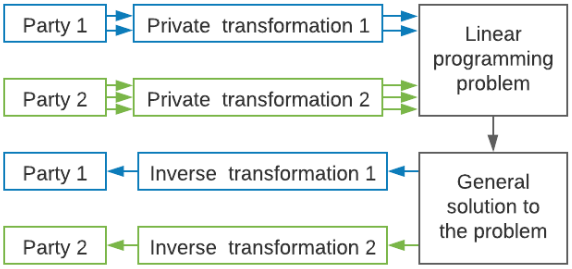

Figure 2: Visualization of secure transformation (two party case)

3 Methodology

In order to discuss recent developments related to secure transformation, it is necessary to formally introduce the methods for collaborating in supply chains, and thus also consider other techniques, albeit in less detail, for a complete overview. To this end, a literature review will be conducted, with a heavier focus on secure transformation, but recent developments for MPC and trusted third party will also be considered, in order to give a proper comparison of how collaborative techniques evolved in recent times. This method was chosen, since the non-cryptographic approach of secure transformation is a relatively new development in the field of collaborative computation and it has not been deployed in real life applications. Unlike MPC, which has been under development for at least 35 years (one of the earliest breakthroughs being in 1986 by Yao [3]), secure transformation has been seeing interest for about two decades, and while this produced many valuable results, they were only theoretical. Thus, a literature review of the techniques and approaches is the most suitable method for producing a relevant overview of developments in secure transformation.

As the focus of this paper is the secure transformation, historical developments (during 2009-2014) for this approach, mainly linear programming which uses secure transformation, are presented in a following section. These developments include the ones mentioned in [1] and [7]. Then, recent developments (later than 2014) for all techniques are discussed, namely [8–11] (secure transformation), [12–14] (MPC), and [15,16] (trusted third party). The section concludes with a comparison of the developments.

A final section considers real-life applications such as scheduling and routing [17], container logistics [18], and airline management [19]. The feasibility of using secure transformation or other techniques (in order to protect the privacy of the mentioned functions) is discussed.

It is worth noting that most of the mentioned literature does not specifically refer to supply chain collaboration and instead features different collaborating parties for which the technique is developed. Regardless, the techniques are, in general, applicable in the context of collaborators in a supply chain who want to avoid sharing their private information while solving some collaborative linear programming problem (for example, minimizing the cost of distributing goods via trucks).

4 Literature Review of Privacy-Preserving Collaborative Techniques

This section presents one of the main contributions of this work, namely the review of existing work. The relevance of linear programming in supply chain is presented, then developments are discussed, and a comparison of how the techniques developed is also provided. The papers discussed throughout this section include the ones mentioned in [1] and [7], as well as in many other related papers.

4.1 Relevant Terms and Definitions

The relevance of linear programming in supply chain is described in [1]. Among the applications, the authors mention minimizing transportation costs, maximizing manufacturing profit, and optimally assigning of the crew in the airline industry, with the goal of minimizing costs while satisfying constraints. The relevance or arbitrarily partitioned linear programs is mentioned in [7], along with the high frequency of encountering this situation in supply chains. The example they give is that of multiple companies that produce wine at multiple wineries, with grapes from multiple vineyards which have a limited production. These companies want to collaborate to reduce the costs of transporting the grapes to the right winery, as the wineries each require specific types of grapes, and they do so through solving an arbitrarily partitioned linear programming problem. Thus, the general privacy issue that secure transformation tries to solve is preserving the private data of the collaborators who are solving a linear programming problem.

Another important term that needs to be defined is the general model of a linear programming problem. According to [20], linear programming problems represent a subset of optimization problems. Furthermore, optimization problems are defined as follows: "given a function f : A $\to$ R from some set A to the real numbers, we seek an element x_0 in A such that f(x_0) $\leq$ f(x), $\forall$ x $\in$ A ('minimization') or such that f(x_0) $\geq$ f(x), $\forall$ x $\in$ A ('maximization')." [20, p. 4] and the main components are: "An objective function which we want to minimize or maximize", "A set of unknowns or variables which affect the value of the objective function", and "A set of constraints that allow the unknowns to take on certain values or exclude others" [20, p. 5]. The defining aspect is the fact that the constraints and the objective function are linear.

It is relevant to also define the possible ways of partitioning the constraints among the participating parties: horizontal partitioning (a constraint belongs exclusively to one collaborator), vertical partitioning (constraints belong to multiple collaborators), and arbitrary partitioning (a combination of the previous two) [20]. Note that these terms are different from horizontal and vertical collaboration, which refers to the roles of the collaborators withing the supply chain.

Lastly, [11] defines the "canonical form" of a linear programming problem as "maximize $c^T \cdot x$, subject to $Ax \leq b, x \geq 0$" [11, p. 218], while the "standard form" is "maximize $c^T \cdot x$, subject to $Ax = b, x \geq 0$" [11, p. 218], where the equality is gained through the introduction of "slack variables" (which allow the inequalities to happen in the end), and the "feasible region" of the problem's solution is the set of all of its solutions, which are vectors.

4.2 Historical Developments for Secure Transformation

Vaidya's "Privacy-Preserving Linear Programming"

Vaidya's paper from 2009 [20] focuses on solving a linear programming problem with two collaborating parties, through secure transformation. The paper considers the case of an arbitrary partitioning: one party (P$_1$) has the objective function , while the other (P$_2$) has the set of constraints. The party which has the cost function is the one which solves the problem. The two party problem is described as: "

$$\begin{aligned} &\max\, (\mathbf{c}_1 + \mathbf{c}_2)^T \mathbf{x} \\ &s.t.\ (M_1 + M_2)\mathbf{x} \leq (\mathbf{b}_1 + \mathbf{b}_2) \\ &\qquad\qquad\qquad \mathbf{x} \geq 0, \end{aligned}$$

where **x** is an n-dimensional vector of variables, $\mathbf{c}_1$+$\mathbf{c}_2$ is an *n*-dimensional vector of objective function coefficients, M_1+M_2 is an $m \times n$ matrix of constraint coefficients (where there are *m* constraint inequalities) and $\mathbf{b}_1$+$\mathbf{b}_2$ is an *m*-dimensional vector of constraint values." [21, p. 118]. In [20], $\mathbf{c}$ = $\mathbf{c}_1$+$\mathbf{c}_2$, M = M_1+M_2, and $\mathbf{b}$ = $\mathbf{b}_1$+$\mathbf{b}_2$. The transformation (is based on Du's transformation [22] and) consists of a vector space transformation, of both the input and the problem, through multiplication with a matrix Q with strictly positive elements, and then transforming back by multiplying with the inverse matrix Q^{-1}. Additionally, the paper notes that secure transformation relies on matrix multiplication, which consists of many dot products. In order to do these computations efficiently, they use the provably secure solution proposed by Goethals et al. in 2004 [23], which uses Homomorphic Encryption.

Later in the same year, Bednarz presents an issue with this transformation in [21]. This is the fact that the modified problem has solutions that are invalid for the original problem, due to the constraint of the positive matrix elements, which in turn cause some elements in the inverse matrix to be negative and flip inequality symbols (the same issue applying to Du's original transformation in [22]). The authors also present a solution to this problem: allowing the matrix to have zeroes and using "generalized permutation matrices", with the downside that these provide less security as P$_1$ can try all permutations of this matrix (to see which one produces the correct final result and thus) to deduce P$_2$'s private data.

In a 2013 paper, Hong and Vaidya introduce a new transformation in [24], that corrects the original issue by using the matrix suggested in [21]. The privacy issue is solved through the following changes: the final result is partitioned, with these partitions being kept private, so that the initial matrix cannot be guessed, and $\mathbf{c}^T$ is also partitioned into $\mathbf{c}_1{}^T$ and $\mathbf{c}_2{}^T$, with them being permutated as well. The transformed vector $\mathbf{c}^T$Q is also kept private, to prevent deducing Q from $\mathbf{c}^T$ and $\mathbf{c}^T$Q. Furthermore, this revised transformation can be applied to the multi-party (more than two parties) case, where each party *i* holds a share M$_i$ of M and a share c$_i^T$ of c^T.

Mangasarian's "Privacy-preserving horizontally partitioned linear programs"

Mangasarian also tackles the multi-party situation. He presents two transformations: one for vertically partitioned linear problems [25], and one for horizontally partitioned ones [26].

The vertical problem is described as:"

$$\min_{x \in X} c'x \text{ where } X = \{x \mid Ax \geq b\},$$

and the matrix $A \in R^{m \times n}$ together with the cost vector $c \in R^n$, that is $\left[\begin{smallmatrix} c' \\ A \end{smallmatrix}\right]$, are divided into *p* vertical blocks of $n_1, n_2, \ldots\ldots$ and n_p, (*m*+1)-dimensional columns with $n_1 + n_2 + \cdots + n_p = n$. Each block of columns of A and corresponding block of the cost vector c are 'owned' by a distinct entity that is unwilling to make this block of data public or share it with the other entities." [25, p. 166]. Note that c$'$x here represents scalar (inner) product and c$'$ is the row representation (transpose) of vector c. For this vertical case, the transformation consists of each party j multiplying its owned column A$_{\cdot j}$ and cost vector element c$_{\cdot j}$ with a random (*m*+1)-dimensional row vector B$_{\cdot j}{}'$. Therefore, the public information accessible to all parties is the matrix BA (where B and A are formed by the row and column vectors respectively) and the vector Bb.

For the horizontal case, the problem and transformation are similar, the only differences being the fact that $X = \{x \mid Ax = b, x \geq 0\}$ and rows and columns are swapped throughout the problem (i.e. parties now own rows and pick a column vector B$_{\cdot I_i}$). It is also mentioned that the same method cannot be used if the equality constraint is turned into an inequality, since the transformation would not "preserve the original feasible region of the problem. Furthermore, if we convert the inequality constraints to equality constraints by adding slack variables, multiplying the *i*th identity matrix coefficient matrix of the slack variables of the *i*th entity by its privately held random matrix B$_{\cdot i}$ would reveal B$_{\cdot i}$" [26, p. 435].

The horizontal transformation is then extended to inequality constraints by Li et al. in [27]. The paper avoids the issue pointed out by Mangasarian by converting the problem from its original form to:"

$$\begin{aligned} &\min c^T x \\ &\; s.t.\ Ax + Dx_s = b,\ x, x_s \geq 0.\text{ "}[27,\ p.\,140] \end{aligned}$$

Here, each party *i* ($1 \leq i \leq p$) chooses a random diagonal matrix D_{I_i} with positive entries, forming the diagonal matrix $D = diag(D_{I_1}, D_{I_2}, \ldots, D_{I_p})$. Also, x_s is the vector of slack variables $x_s = (x_{m+1}, x_{m+2}, \ldots, x_{m+n})^T$. Rather than just adding slack variables to accommodate inequalities, they also multiply them with random positive numbers, so that the private matrix B$_{\cdot i}$ is not revealed.

The next year, Hong and Vaidya point out a possible inference attack to both [26] and [27] and revise the transformation in [28]. The attack is formulated in terms of *m*, the number of constraints of the linear programming problem: by learning *m*, an attacker could "infer other entities' private constraints by formulating equations with real variables" [28, p. 270]. The paper states that *m* could be discovered as either the number of slack variables *l* (which is made public while solving the problem), or as the rank of the matrix A (which can be found from the public information about the transformation, more specifically, from the rank of the matrix BA), which is often equal to the number of constraints *m*. To tackle the latter issue, artificial inequality constraints are generated, in order to increase the rank of the matrix A, and thus protect the rank of A from being inferred (the only property that can be inferred now is *m'*, the number of constraints of the artificially-extended problem). The solution to the former issue also relies on these artificial constraints, more specifically on the fact that by the number of slack variables has to increase, in order to accommodate the newly added constraints. The authors also mention that the chance of inferring *m* can further be reduced by adding multiple slack variables when converting inequality constraints to equality constraints.

It is worth mentioning the applications of these papers in supply chain. According to [1], the horizontal partitioning present in [26] and [27] ”can be utilized to secure the production process in which each factory privately holds a different kind of raw material, or secure the task machine scheduling process in which every machine is held by one party” [1, p. 256], while the vertical partitioning in [25] ”could secure the transportation in which all companies share their trucks to bound the shipping” [1, p. 256].

Secure Outsourcing of Computation Through Transformation

In their 2011 paper [6], Dreier and Kerschbaum introduce a variant to Vaidya’s transformation in [20] and one of Mangasarian’s older transformations (from 2010) in the context of secure outsourcing of the problem to the cloud. In the context of supply chain collaboration, the cloud could be seen as an untrusted third party that the collaborators use to solve the distributed problem. In this case, the constraints would be partitioned across all collaborators. Outsourcing problem computation to the cloud is regarded as an easy way of gaining access to more computational resources, and this also applies to the supply chain case. The paper claims that Vaidya’s work had security issues and Mangasarian’s does not cover problems often encountered in supply chains, while both of them lack a security analysis. The transformation they use is very similar to Vaidya’s transformation in [20], with he difference being that a positive vector **r** is used to hide the vector of variables **x**. As a result, they claim that the transformation is correct and prove this claim, and, furthermore, they show that its efficiency is is better than that of cryptography-based methods from that time. The security of the transformation is thoroughly analysed through ”Leakage Quantification” methods, and the authors also perform several experiments with a simulated supply chain, to show that the transformation is efficient and that data leaks are minor.

Outsourcing of computation to the cloud is also investigated by Wang et al. in [29]. The cloud is considered to be capable of solving a general linear programming problem, and the private data (the problem coefficients) is first transformed locally and then sent to the cloud for the computation. The communication relies on encryption of the data sent, through a randomly generated key K = (**Q**, **M**, **r**, λ, γ). This key is composed of the transformation elements: a random matrix **Q**, which is used to hide the equality constraints of the problem, a random matrix λ, which is used to hide the equality constraints, and a scalar value γ, used for hiding the objective function. Furthermore, an affine mapping (represented by multiplying with a random non-singular matrix **M** and adding a random vector **r**) is used to hide the feasible region of the original problem and the output. The hidden (transformed) problem is denoted as Φ_K. Note that this approach is different from the ones encountered so far, since none of the transformations used separate matrices to encrypt different parts of the problem, and most of them only used affine transformations to hide the feasible region of the problem. In doing so, this paper enhances the security of their protocol. The paper also develops an anti-cheat system (a way to check of the answer given by the cloud is actually correct). The method employed is different for each type of problem (normal - there is a solution, infeasible - there is no solution, unbounded - the solution is infinitely large), but it is shown that the overhead is always low (in theory), for both customers and the cloud. Through an experiment, this low overhead is proven to be correct, and furthermore, it is shown that the computation for normal problems is at least roughly 25 times faster (with the maximum being about 47 times faster) when done on the cloud than what it would have taken locally.

Other Work

Weeraddana et al. also partially discuss secure transformation in the 2012 paper [30]. Rather than covering a specific problem, this work looks at the general approaches of secure transformations and categorizes them into one of two classes: change of variables and transformation of objective and constraint functions. For the change of variables, the transformation can be scaling, translation, affine transformation, or a non-linear transformation. As it turns out, most the papers discussed thus far ([6, 20–22, 25, 29]) all provide an affine transformation. For a transformation of objective and constraint functions, scaling and partitioning the problems are techniques for preserving privacy. For both classes of transformations, the paper provides a proposition for the general form of the transformation and the privacy guarantees that it provides, along with generalized proofs for the privacy guarantees. The authors note that hybrid transformations also exist, where both kinds of transformations are applied sequentially, and the example given is that of [6], where there is first a change of variable and then a scaling transformation.

4.3 Recent Developments

As was seen in the previous subsection, transformations are difficult to formulate, as privacy vulnerabilities might be initially overlooked. This subsection presents recent developments for secure transformation, and the other techniques presented in 2, for a full overview of recent work in the field of privacy-preserving collaboration.

Secure Transformation

In 2015, Pankova and Laud take an in-depth look at the technique of secure transformation when applied to linear programming problems in [11]. The paper provides attacks to a general form of the existing transformations at that time, and tries to provide definitions that prove that the security is comparable to that of a cryptographic approach. Unfortunately, their attempts at cryptography-based transformations only prove that transformation approaches cannot reach information-theoretical security. The authors doubt the existence of transformations which achieve computational security, but they mention that there are cases with provable security, and the example given is that of working within a finite field to outsource the computation of a matrix inverse. Their conclusion is that ”cryptography over real numbers has not received much attention so far. Extending finite field assumptions to real numbers is not possible in general.” [11, p. 244].

The most recent version of the original transformation proposed by Vaidya in 2009 ([20]) is [7]. The techniques previously described in [24] remain close to identical, save for a few corrections in the procedures. This indicates that the open questions posed in both [24] and [7] are still unanswered. Among those, the security of transformations ”is still somewhat heuristic” [7, p. 14]. They argue that, while the original data is kept secure, as this is the purpose of the transformation, it is not clear whether or not the transformation reveals links between the constraints or the difficulty of the original problem. Additionally, integer programming and quadratic programming are mentioned as candidates for good transformations.

A recently published paper by Zhang et al. [8] is related to Mangasarian's work from 2012 ([26]). Specifically, this paper mentions that an issue with the original transformation is that the random matrix used is not always full rank, and in such cases, the transformed problem is not the same as the original one. They solve this problem by using an invertible matrix that they multiply with the two sides of the equality constraints.

The linear problem considered is:”

$$\min z = c^T x$$
$$\text{s.t. } Ax = b$$
$$x \geq 0.$$

Here, $(A \quad b)$ consists of the matrix $A \in R^{m\times n}$ and the right-hand vector $b \in R^m$ and is divided into *p* horizontal blocks. The number of rows of the *p* horizontal block is recorded as m_1, m_2, $\ldots$, m_p, where m_1 + m_2 + $\cdots$ + m_p = *m*. An *m* order identity matrix E is divided into *p* vertical blocks. The number of columns of the *p* vertical block is recorded as m_1, m_2, $\ldots$, m_p, where m_1 + m_2 + $\cdots$ + m_p = *m*. Each block of rows of $\left[\, {\scriptstyle A\ b} \right]$ corresponding to the index sets I_1, I_2, $\ldots$, I_p, $\cup_{i=1}^{p} I_i$ = $\{1, 2, \ldots, m\}$, is owned by a distinct entity that is unwilling to make its block of data public or share it with the other entities.” [8, p. 1]. In order to do the transformation, the parties choose a number $\lambda \geq n$ together, then each party *i* ($1 \leq i \leq p$) computes its own private matrix $B_{.I_i} \in R^{m\times m_i}$ with elements in (0,1). The matrix $B = \left(B_{.I_1} + \lambda E_{.I_1} \quad \mathrm{B}_{.I_2} + \lambda E_{.I_2} \quad \cdots \quad \mathrm{B}_{.I_p} + \lambda E_{.I_p}\right)$ is then used as the transformation. The paper argues that, since this matrix is invertible, the solution of the transformed problem is the same as the original one, and, while this solution is public, it does not reveal any information about the private data.

The authors of [29], improve their original paper in [9]. The main addition in this paper is the investigation regarding the connections between the original (Φ) the the transformed problems (Φ_K). Through this, the authors conclude that two new problems, Ψ and Ψ_K, derived from Φ and Φ_K respectively (by multiplying their constraint matrices and result vectors), also share the same feasible region. Additionally, the enhancements for the hiding of the feasible region of the problem are also investigated in detail, with the result being that if the matrix used to achieve this happens to be the identity matrix **I**, then the feasible regions are the same, and this poses a security threat. The solution proposed is to also multiply with another matrix **P**, which needs to be a generalized permutation matrix with positive non-zero elements. Furthermore, the original experiments are extended to also cover infeasible and unbounded problems, and performed again for normal problems. The results show that the speedups (the time it would have taken to solve the problem locally, divided by the time it takes to solve it on the cloud) for normal problems are, in fact, at least 50 times faster (and as much as 434 times faster) when done on the cloud. For infeasible problems, speedups are at least 39 times and at most 364 times, while for unbounded problems, they are at least 68 times and at most 497 times.

Two years later, Li et al. present developments for outsourcing non-linear problems to the cloud in [10]. The reasoning for outsourcing to the cloud are the same as the ones discussed for [29], but as far as (integer) non-linear programming is involved, examples of its relevance are in the citrus supply chain ([31], [32]), but also transport and logistics ([10]). The problem considered can be the same one as the one in [29], but some of the transformations provided are different. Equality constraints are hidden by multiplying with two matrices **P** (random diagonal matrix) and **Q** (positive constant diagonal matrix), while inequality constraints are hidden through multiplying with matrices **T** (random diagonal matrix) and **S** (positive constant diagonal matrix). Similar to the approach of [29], the output is protected by adding a random vector **r**. An experiment provides insight into the efficiency of the protocol, with a minimum speedup of 34 times faster and a maximum of 49 times faster. Since this is the case, it also means that the overhead gained from outsourcing the computation to the cloud is quite low.

MPC

Some recent developments of MPC applications that have been deployed in practice are presented in by Lindell in [12]. Among these, the most relevant one for supply chain is cryptographic key protection, through a technique called threshold cryptography, which ”provides the ability to carry out cryptographic operations (like decryption and signing) without the private key being held in any single place” [12, p. 13].

The author of [12] also mentions [13], which is a book by Evans et al., dedicated to secure multi-party computation. Aside from explaining the many techniques (from garbled circuits to oblivious transfer, in chapter 4) in detail and presenting secure techniques for dealing with malicious adversaries (from cut-and-choose to authenticated garbling, in chapter 6), the book also mentions recent developments in the field. For example, chapter 4.1.3 talks about less expensive garbling, through half gates. This is achieved through a garbling technique developed by Zahur et al. [33] which rewrites an AND gate as an exclusive or of two "half gates" ("which are AND gates where one of the inputs is known to one of the parties" [13, p. 69]). The two needed gates are called generator and evaluator half gates. As another example, chapter 7.1.1 mentions more developments on garbled circuits such as the ones by Mohassel et al. [34], who present a 3-party variant of the original garbled circuits. Their approach is to use two parties as circuit garblers and the third party as an evaluator, who only needs to verify that the two generated circuits are identical. The book considers this to be secure under an honest majority assumption (at most one of the parties is corrupted by an adversary). Furthermore, it is mentioned that Patra and Ravi [35] have enhanced this protocol with "fairness (if the adversary learns the output, then the honest parties do) and guaranteed output delivery (all honest parties will receive output)" [13, p. 128], and that Chandran et al. [36] extend it to the case of n parties, out of which roughly $\sqrt{n}$ are corrupted.

Lastly, in [14] Bayatbabolghani and Blanton present several references to recently developed compilers for MPC, among which ObliVM [37], and Obliv-C [38].

Trusted Third Party

A recently developed technique for collaborating using a trusted third party is introduced by Tueno et al. in [15]. As an item passes through different components of the supply chain, data such as "time, location, and type of handling (e.g., packing, unpacking, receiving, or shipping)" [15, p. 476] is gathered by using Radio Frequency Identification (RFID) tags. The paper mentions how this data can be used for collaborative applications such as estimated arrival forecasts, counterfeit detection, batch recalls or analytics. As is often the case in collaboration, some data might need to remain private. Thus, their approach is to store the data in a cloud that all the collaborators can access, and encrypt parts of the data that are considered private (selective encryption), through a public key infrastructure offered by a trusted third party. The trusted third party is responsible for distributing the RFID tags and for initializing them with private signatures. The collaborators encrypt their data using different keys, which are shared only with parties that should have access to the data. In this way, even the cloud provider does not have access to the private data, as the keys are shared through the trusted third party.

While this paper does not cover any collaborative application specifically, by enabling the collaborators to share their data with each other through a trusted third party, it paves the way for solving a collaborative problem. It is worth nothing, though, that this approach wanders away from the initially discussed role of the trusted third party, namely that of actually computing the function, not providing a means to share data between collaborators who trust each other.

Comparison of Developments

Recent literature suggests that MPC is the collaborative technique which has seen the most improvement recently. While the transition from a theoretic concept to an approach used in practice is not something that happened recently, newer developments help push its adoption as a solid and provably secure strategy, due to it being based on cryptographic properties.

Trusted third party has also seen developments, but there also seems to be a different direction for this approach. More specifically, trusted third parties are used less as a party which is given all the private information and tasked with finding the optimal solution, and more as a mediator which allows parties themselves to collaboratively solve the problem. This can be seen not only in the paper mentioned before ([15]), but also in papers such as [16] (where third parties are used to from collaborative relationships).

While some developments have been published for secure transformation, the lack of a good way of proving the security of the protocols, and having to rely on heuristic analysis still seems to prevent it from becoming a technique used in practice. While the work of [11] raises doubts towards the feasibility of using secure transformations for linear problems using real numbers, the technique as a whole is not condemned to failure. Linear programming is not the only type of problem found in supply chains and theoretical developments for particular cases of problems are being made. Research towards non-linear programming (and future research into real number cryptography-based transformations) also opens possibilities for secure transformation to be deployed in supply chain applications which value computational efficiency over the limitation of only "heuristic" privacy.

Table 1, listing all of the developments discussed thus far can be found in Appendix 1.

5 Real-life Applications and Limitations

This section provides a look at a few different collaborative tasks, and constitutes the second part of the contribution. A short description, about what the task implies and what its privacy requirements are, is given, and the feasibility of using a secure transformation approach to accomplish this is evaluated. Limitations of secure transformation are also discussed.

Collaborative Scheduling and Routing

Two of the problems discussed in [17] are last mile delivery (meaning the last step of delivery to the actual customer) and bin packing (packing items such that the number of bins is minimized), both of which are relevant in collaborative supply chains. The paper mentions that there are linear programming approaches to these problems, namely [39] (which presents the problem as a set partitioning problem, for which [40] presents a linear programming approach) and [41] respectively. Given this fact, depending on how data is partitioned among the collaborators, various secure transformations could be used. For example, if there is a horizontal partitioning, approaches such as the ones in [26] or [8] are suitable. For vertical partitioning, [25] presents a solution, while [7] works for any arbitrary partitions.

Collaborative Container Maritime Logistics

The situation of collaboration in maritime transport, more specifically focusing on containers, is presented in [18]. This is done between ports and port users, whose main collaboration interest is through information sharing. If the security of the information is a concern, the approach in [15] could involve a third party that facilitates secure communication. Another aspect of collaboration within this field is joint supply chain performance measurement, which is concerned with ”optimising related port activities including container handling time, number of vessels to be accommodated, port time, berth utilisation, and joint actions in security and risks” [18, p. 300]. As such, optimization problems could be formulated (perhaps even as linear programming problems in some cases) and solved in a privacy-preserving manner through secure transformation.

Collaborative Airline Management

Within airline management, [19] identifies schedule design as one of the most important decisions. As airlines are usually part of airline alliances, the individual decisions also depend on the collaborators' opinions, with the main goal of making efficient use of all the resources available. In this manner, optimization problems could be formulated, and, as was the case for collaborative container management, some of them could be linear programming problems. Normally this would imply that secure transformation is a valid strategy, however, since they form an alliance, these collaborators are likely to trust each other more compared to other supply chains. If this is the case, there is no need to protect information from other parties, but there is still a need for securely sharing this information, and then a third party can facilitate this [15]. If, however, an alliance decides to collaborate with a different one, the usual case applies, and secure transformation techniques can be employed, with the same discussion regarding partitioning as for scheduling and routing. As schedule design is a difficult problem, perhaps outsourcing the problem to a cloud with more computation power, as mentioned in [9] and [10], is also a possible approach.

Limitations

The main limitations of the technique of secure transformation come from its origin as a non-cryptographic approach. As [11] discusses, if perfect secrecy is the goal of the protocol, secure transformation is infeasible and techniques such as MPC are more suitable. That is not to say that using secure transformation is the same as having no security, but, given an adversary with enough computational resources, data regarding the original problem might leak. If this data happens to be highly private information, then this poses a major privacy issue. The most important challenge, for any supply application which wants to employ secure transformations, is to figure out whether or not efficiency is valued over security.

6 Responsible Research

First of all, as this work features only a literature review, no experiments were performed as part of the research. The sources themselves are also mostly theoretical (some also being literature reviews), featuring little to no experiments. Regardless, the experiments that do appear feature randomly generated matrices, which could (theoretically) be cherry picked to feature better results. However, that would not affect the result, as the random numbers themselves do not influence the security of the protocol, and are only meant to provide data obfuscation.

Secondly, the papers studied and the information presented were also not specifically selected to show only the positives of secure transformation. While works such as [1] and [7] mention just some downsides of secure transformation, [11] criticizes the approach as a whole and shows why it is unsuitable for providing privacy when working outside a finite field.

Thirdly, all the information taken from a paper written by a different author is cited, and the references provided are given for the correct papers, with correct information. While [1] was used as a basis for definitions, and papers cited by this source were researched in detail and explained here, their work is credited wherever this is the case and not closely replicated in content, nor in structure.

Lastly, the research performed is highly reproducible, given that it is a literature review (which includes all of the sources used), and conclusions should mostly align with the following statement: the technique can work in supply chains, but it has obvious flaws which need to be taken into account in order to minimize the risk of leaking private data.

7 Conclusions and Future Work

In this paper, the question of how secure transformation preserves privacy in collaborative supply chains was tackled. To this end, the collaborative technique of secure transformation was investigated in detail. Both older and recent papers highlight the efficiency of this approach when compared to cryptographic approaches such as secure multi-party computation (MPC). Secure transformation preserves the privacy of collaborative supply chains by transforming the original input and problem. Such a transformation is usually composed of one or more matrix multiplications, and the most common types of problems that it is applied to are linear programming problems, where two or more parties try to collaboratively solve a problem without revealing private data. This situation can often be found in a real life supply chain, and thus this model generally fits the description and privacy needs of a collaborative supply chain.

Despite its positives, the general consensus is that this technique is not perfect, as it has the downside of ”somewhat heuristic” privacy [7, p. 14] or even its impossibility of reaching the same security as a cryptographic approach when working outside a finite field [11]. Being a non-cryptographic technique, the security of the protocols cannot be proved through well known and properly understood cryptographic primitives. Thus, issues might arise, either in having an erroneous transformation that covers a larger solution (i.e. some solutions to the transformed problem are not solutions to the original problem), in the difficulty of coming up with a proper proof for the privacy, or even in the impossibility of formulating a privacy-preserving transformation at all.

Other collaborative techniques, namely trusted third party and MPC were defined in the introduction, and recent developments for them were also discussed. In comparison to secure transformation, the reliability of MPC in terms of privacy preservation seems to have pushed the approach into use for real life applications (though not necessarily in the supply chain yet). It appears that, in many cases, being able to prove that the data is indeed private, through cryptographic means, is favourable over speed.

The purpose of secure transformation in supply chains is, however, not missing. It is a technique for which much theoretical research is still being conducted. The broad area of linear programming has applications in fields such as logistics and scheduling, for both land and maritime supply chains. Recent research into non-linear programming (which also has some applications in the supply chain) has also been conducted for cases of outsourcing the computation to an untrusted cloud. Older transformations are being revised and fixed, and transformations for certain problems and data partitionings which are encountered in supply chains are still being developed. Additionally, more research into cryptography over real numbers is a much-needed tool for a better chance of finding a secure transformation.

To conclude, the technique is still advancing, and some questions regarding it still remain. Among the most crucial is whether or not the trade-off between efficiency and privacy is suitable for a real-life supply chain. Unlike the theory-based simulations ran so far (which proved this to be the case), a data leak in the real world could produce heavy losses, and therefore the first real experiment should be performed with this aspect in mind. Another question is whether or not a solid non-linear transformation is possible. Though research has shown that there are cases where secure transformations for linear problems are infeasible, perhaps a non-linear transformation specifically targeted at supply chain problems and their security necessities could be developed. By rigorously studying and modifying it to ensure both a privacy akin to that of a cryptographic approach and the efficiency of the non-cryptographic one, there would be no need for a trade-off. A candidate for such a transformation has already been provided in [10], but it only tackles computation outsourcing and rigorous research and real-life applications have not been investigated yet.

References

[1] Y. Hong, J. Vaidya, and S. Wang, "A survey of privacy-aware supply chain collaboration: From theory to applications," *Journal of Information Systems*, vol. 28, no. 1, pp. 243–268, 2014.

[2] L. Chen, X. Zhao, O. Tang, L. Price, S. Zhang, and W. Zhu, "Supply chain collaboration for sustainability: A literature review and future research agenda," *International Journal of Production Economics*, vol. 194, pp. 73–87, 2017. Special Issue: Innovations in Production Economics.

[3] A. C.-C. Yao, "How to generate and exchange secrets," in *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pp. 162–167, IEEE, 1986.

[4] S. Micali, O. Goldreich, and A. Wigderson, "How to play any mental game," in *Proceedings of the Nineteenth ACM Symp. on Theory of Computing, STOC*, pp. 218–229, ACM, 1987.

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)," pp. 1–10, 01 1988.

[6] J. Dreier and F. Kerschbaum, "Practical privacy-preserving multiparty linear programming based on problem transformation," in *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pp. 916–924, IEEE, 2011.

[7] Y. Hong, J. Vaidya, N. Rizzo, and Q. Liu, "Privacy preserving linear programming," *CoRR*, vol. abs/1610.02339, 2016.

[8] C. Zhang, D. Kong, P. Pan, and M. Zhou, "A new algorithm for privacy-preserving horizontally partitioned linear programs," *Journal of Mathematics*, vol. 2021, 2021.

[9] C. Wang, K. Ren, and J. Wang, "Secure optimization computation outsourcing in cloud computing: A case study of linear programming," *IEEE transactions on computers*, vol. 65, no. 1, pp. 216–229, 2015.

[10] A. Li, W. Du, and Q. Li, "Privacy-preserving outsourcing of large-scale nonlinear programming to the cloud," in *International Conference on Security and Privacy in Communication Systems*, pp. 569–587, Springer, 2018.

[11] A. Pankova and P. Laud, "Transformation-based computation and impossibility results," in *Applications of Secure Multiparty Computation*, pp. 216–245, IOS Press, 2015.

[12] Y. Lindell, "Secure multiparty computation (MPC).," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 300, 2020.

[13] D. Evans, V. Kolesnikov, and M. Rosulek, "A pragmatic introduction to secure multi-party computation," *Foundations and Trends® in Privacy and Security*, vol. 2, no. 2-3, 2017.

[14] F. Bayatbabolghani and M. Blanton, "Secure multi-party computation," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2157–2159, 2018.

[15] A. Tueno, F. Kerschbaum, D. Bernau, and S. Foresti, “Selective access for supply chain management in the cloud,” in *2017 IEEE Conference on Communications and Network Security (CNS)*, pp. 476–482, IEEE, 2017.

[16] B. Pinnington, A. Lyons, and J. Meehan, “Orchestration of business collaboration by third-party brokers,” 2017.

[17] S. S. Azadeh, Y. Maknoon, J. Chen, and M. Bierlaire, “The impact of collaborative scheduling and routing for interconnected logistics: A european case study,” in *Strategic Decision Making for Sustainable Management of Industrial Networks*, pp. 35–56, Springer, 2021.

[18] Y.-J. Seo, J. Dinwoodie, and M. Roe, “Measures of supply chain collaboration in container logistics,” *Maritime Economics & Logistics*, vol. 17, no. 3, pp. 292–314, 2015.

[19] M. Sigala, “Collaborative supply chain management in the airline sector: the role of global distribution systems (GDS),” in *Advances in Hospitality and Leisure*, Emerald Group Publishing Limited, 2005.

[20] J. Vaidya, “Privacy-preserving linear programming,” in *Proceedings of the 2009 ACM symposium on Applied Computing*, pp. 2002–2007, 2009.

[21] A. Bednarz, N. Bean, and M. Roughan, “Hiccups on the road to privacy-preserving linear programming,” in *Proceedings of the 8th ACM Workshop on Privacy in the Electronic Society*, pp. 117–120, 2009.

[22] W. Du, *A study of several specific secure two-party computation problems*. PhD thesis, Purdue University, 2001.

[23] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikäinen, “On private scalar product computation for privacy-preserving data mining,” in *International Conference on Information Security and Cryptology*, pp. 104–120, Springer, 2004.

[24] Y. Hong and J. Vaidya, “Secure transformation for multiparty linear programming,” *Rutgers Technical Report*, 2013.

[25] O. L. Mangasarian, “Privacy-preserving linear programming,” *Optimization Letters*, vol. 5, no. 1, pp. 165–172, 2011.

[26] O. L. Mangasarian, “Privacy-preserving horizontally partitioned linear programs,” *Optimization Letters*, vol. 6, no. 3, pp. 431–436, 2012.

[27] W. Li, H. Li, and C. Deng, “Privacy-preserving horizontally partitioned linear programs with inequality constraints,” *Optimization Letters*, vol. 7, no. 1, pp. 137–144, 2013.

[28] Y. Hong and J. Vaidya, “An inference–proof approach to privacy-preserving horizontally partitioned linear programs,” *Optimization Letters*, vol. 8, no. 1, pp. 267–277, 2014.

[29] C. Wang, K. Ren, and J. Wang, “Secure and practical outsourcing of linear programming in cloud computing,” in *2011 Proceedings Ieee Infocom*, pp. 820–828, IEEE, 2011.

[30] P. C. Weeraddana, G. Athanasiou, M. Jakobsson, C. Fischione, and J. Baras, “Per-se privacy preserving distributed optimization,” *arXiv preprint arXiv:1210.3283*, 2012.

[31] N. Sahebjamnia, F. Goodarzian, and M. Hajiaghaei-Keshteli, “Optimization of multi-period three-echelon citrus supply chain problem,” *Journal of Optimization in Industrial Engineering*, vol. 13, no. 1, pp. 39–53, 2020.

[32] M. Fakhrzad and F. Goodarzian, “A new multi-objective mathematical model for a citrus supply chain network design: Metaheuristic algorithms,” *Journal of Optimization in Industrial Engineering*, vol. 14, no. 2, pp. 111–128, 2021.

[33] S. Zahur, M. Rosulek, and D. Evans, “Two halves make a whole,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 220–250, Springer, 2015.

[34] P. Mohassel, M. Rosulek, and Y. Zhang, “Fast and secure three-party computation: The garbled circuit approach,” in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 591–602, 2015.

[35] A. Patra and D. Ravi, “On the exact round complexity of secure three-party computation,” in *Annual International Cryptology Conference*, pp. 425–458, Springer, 2018.

[36] N. Chandran, J. A. Garay, P. Mohassel, and S. Vusirikala, “Efficient, constant-round and actively secure MPC: beyond the three-party case,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 277–294, 2017.

[37] 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*, pp. 359–376, IEEE, 2015.

[38] S. Zahur and D. Evans, “Obliv-C: A language for extensible data-oblivious computation.,” *IACR Cryptol. ePrint Arch.*, vol. 2015, p. 1153, 2015.

[39] F. Massen, Y. Deville, and P. Van Hentenryck, “Pheromone-based heuristic column generation for vehicle routing problems with black box feasibility,” in *Integration of AI and OR Techniques in Contraint Programming for Combinatorial Optimzation Problems* (N. Beldiceanu, N. Jussien, and É. Pinson, eds.), (Berlin, Heidelberg), pp. 260–274, Springer Berlin Heidelberg, 2012.

[40] M. Diaby, “Linear programming formulation of the set partitioning problem,” *International Journal of Operational Research*, vol. 8, no. 4, pp. 399–427, 2010.

[41] M. Hifi, I. Kacem, S. Nègre, and L. Wu, “A linear programming approach for the three-dimensional bin-packing problem,” *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 993–1000, 2010.

A Appendix 1

Table 1: An overview of developments for collaborative supply chain techniques

Name	Important aspects	Year, Reference
	Secure transformation	
Transformation-based computation and impossibility results	• secure transformation cannot reach perfect security when working outside a finite field • cryptographic approaches could help, but cryptography over real numbers has not been thoroughly researched	2015, [11]
Privacy preserving linear programming	• the main advantage of secure transformation is its efficiency • the main issue of secure transformation is its heuristic privacy	2016, [7]
A new algorithm for privacy-preserving horizontally partitioned linear programs	• an older transformation is revised, in order to fix underlying security issues • this shows that the problem of not being able to quantify the privacy causes difficulties in coming up with suitable transformations	2021, [8]
Secure optimization computation outsourcing in cloud computing: A case study of linear programming	• outsourcing computation of linear programs to the cloud and preserving the privacy via transformations is gaining popularity • experimental results show the benefit of outsourcing and the fact the the privacy is preserved • a method of preventing the cloud from cheating by skipping computation and giving random results is developed	2015, [9]
Privacy-preserving outsourcing of large-scale nonlinear programming to the cloud	• outsourcing of computation to the cloud is extended to non-linear programs too • the experiments show that the approach is also beneficial for non-linear programs and that privacy is still preserved • relevant for citrus supply chain, for example	2018, [10]
	MPC	
Secure multiparty computation (MPC)	• MPC is seeing deployment in practice (but not supply chain yet) • cryptographic key protection is relevant for supply chain	2020, [12]
A pragmatic introduction to secure multi-party computation	• a very detailed and rigorous work on MPC • discusses recent developments, such as less expensive garbling (through half gates) and extensions of the original two-party garbling problem to three parties	2017, [13]
Secure multi-party computation	• compilers for MPC are being developed	2018, [14]
	Trusted third party	
Selective access for supply chain management in the cloud	• the use of RFID technology for tracking products through the supply chain, enables third parties to act as an intermediary between two collaborators trying to share data with each other securely	2017, [15]
Orchestration of business collaboration by third-party brokers	• shows that the role of third parties is shifting away from being the party which computes the function to that of the party which enables the forming of collaborative relationships	2017, [16]