Research Center
Finance & Information Management

Fraunhofer FIT
Project Group
Business & Information
Systems Engineering

Discussion Paper

A Privacy Preserving Approach to Collaborative Systemic Risk Identification: The Use-case of Supply Chain Networks

by

Tirazheh Zare Garizy, Gilbert Fridgen, Lars Wederhake

in: Security and Communication Networks, July 2018

University of Augsburg, D-86135 Augsburg
Visitors: Universitätsstr. 12, 86159 Augsburg
Phone: +49 821 598-4801 (Fax: -4899)

University of Bayreuth, D-95440 Bayreuth
Visitors: Wittelsbacherring 10, 95444 Bayreuth
Phone: +49 921 55-4711 (Fax: - 844710)

www.fim-rc.de

WI-650

A Privacy Preserving Approach to Collaborative Systemic Risk Identification: the use-case of Supply Chain Networks

Tirazheh Zare Garizy
FIM Research Center
University of Augsburg
Universitaetsstrasse 12, 86159 Augsburg, Germany
Telephone: +49 821 598 - 4816
Fax: +49 821 598 - 4899
tirazheh.zare-garizy@fim-rc.de

Gilbert Fridgen
FIM Research Center
University of Bayreuth
Wittelsbacherring 10, 95444 Bayreuth, Germany
gilbert.fridgen@fim-rc.de

Lars Wederhake (corresponding author)
Project Group Business & Information Systems Engineering of the Fraunhofer FIT University of Augsburg
86135 Augsburg, Germany
lars.wederhake@fit.fraunhofer.de

A Privacy Preserving Approach to Collaborative Systemic Risk Identification: the use-case of Supply Chain Networks

Abstract

Globalization, and outsourcing are two main factors which are leading to higher complexity of supply chain networks. Due to the strategic importance of having a sustainable network it is necessary to have an enhanced supply chain network risk management. In a supply chain network many firms depend directly or indirectly on a specific supplier. In this regard, unknown risks of network's structure can endanger the whole supply chain network's robustness. In spite of the importance of risk identification of supply chain network, firms are not willing to exchange the structural information of their network. Firms are concerned about risking their strategic positioning or established connections in the network. The paper proposes to combine secure multiparty computation cryptography methods with risk identification algorithms from social network analysis to address this challenge. The combination enables structural risk identification of supply chain networks without endangering firms' competitive advantage.

Keywords: Multiparty Computation, Algorithms, Privacy Preservation, Supply Chain Network, Systemic Risk, Risk Management

1. Introduction

In March 2000, a thunderstorm in New Mexico caused a 400-million-dollar loss for the telecommunications equipment firm Ericsson. The fire in a semiconductor plant, a single source key components provider for Ericsson, led to this damage. This loss could have been lower with an appropriate risk management within the supply chain network (SCN) of Ericsson [1].

High complexity of SCNs and steady increase in vulnerability within the SCN are the results of globalization, digitalization, outsourcing and customer or supplier dependencies [2]. The complex structures of SCNs are vulnerable to systemic risk at all scales. Systemic risk is not just the risk of statistically independent failure, but also the risk of failure cascading within the whole interconnected system [3]. This cascading effect impacts the whole system's performance and can lead to irrecoverable value disruptions [4,5]. 54% of firms are either extremely or very concerned about their sustainability performance [6]. Being one of the four emerging issues in global risk [7], it is inevitable to invest in risk management for supply chains. Managers and public policy makers need to identify risks to perform proper risk management and mitigation plans.

Simulation models [8–10], descriptive case studies [11,12], and development of taxonomies of SCNs [13,14] are common research results of the scholars on analysis of SCNs. The embedded positioning of firms within the SCN is important for each firm in the network as well as for the network as a whole. Innovation adoption, influence power or brokering activities of the firms can be derived from their structural positioning in the SCN. Moreover, the structural positioning of the firms can affect the vulnerability or robustness of the SCN [15]. Over the last few decades, the importance of adopting a network perspective in supply chain analysis and management has increased. Recently, the idea of adopting network measures for the investigation of SCNs is opening new potentials to evaluate supply chains [16,16,17].

There are several measures to quantitatively characterize the network structure. Each measure can be adopted to capture a specific feature of the network [18]. Betweenness, closeness, and degree centrality are some of the widely used measures in social network analysis [19,20]. Kim et al. [15] mapped these measures within the SCN and defined their implication for two types of supply networks: material flows and contractual relationships. They identified that firms with higher betweenness centrality (BC) have a higher impact on the product quality, coordination cost, and lead time or can cause unwanted intervene or control among the SCN. These risky firms have a higher contribution to systemic risk. The BC is an indicator for identifying firms with the possibility of influencing information processing, strategic alignments, and perverting risk management within the supply network [15]. Based on Hallikas et al. [21] the risks in

a SCN can affect the long-term sustainable competitive advantage of the network. Considering our focus and above mentioned findings, we assume the BC to be an appropriate measure to identify risky firms in the SCN.

One of the main challenges in studying supply chain risks is the scarcity of real life data on SCNs [15,22]. The fear of risking competitors' advantage by information sharing hinders firms' collaboration within the SCN. To calculate the BC, either based on definition [18,19], or by means of widely used algorithms such as Brandes' [23], having information about the network's structure is necessary. This structural information contains data on the network's firms and their possible connectivity to other firms. However, the strategic importance of the firms' position and connections within the network [24] dissuades firms from sharing this information. In this case, the application of secure multiparty computation (SMC) cryptographic algorithms [25,26] would be one of the solutions to facilitate information sharing willingness within the network. SMC algorithms are based on simultaneous exchanges of encrypted data among parties. The result is calculated from the encrypted data, and is shared among all firms (parties) in the network. The algorithm prevents leakage of key information between the firms.

Summarizing, we find considerable support for the importance of risk analysis in SCNs and the adequacy of the BC to identify the bottlenecks in SCNs. Literature, however, also backs that firms are reluctant to share information on their position in the network. Having this as a starting point, the main focus of this paper is to introduce an artifact – based on the design science paradigm – for privacy preserving calculation of the BC of a given SCN. This paper is an extended version of our prior research [27,28] and includes detailed information on the developed artifact, the pseudocode of the artifact, and a detailed description and explanation of the pseudocode. Our artifact consists of four main methods that are calculating the desired result. The main contributions of our paper are:

- Identification of risks: In the first step of risk management it is necessary to develop models and methods for risk identification in SCNs. In a small SCN, firms are more likely to keep the overview of the SCN topology and the firms in the network. Consequently, in such cases risks are relatively transparent and privacy might not be the main subject of interest. Our concern is the risk identification in large SCNs consisting of hundreds of inter-connected firms. In a large SCN, on the one hand the identification of unknown risks is important and on the other hand the privacy of members should be maintained. For an increasing size of the SCN and the inter-relationships among the firms, the network becomes more complex [29,30]. Due to the higher complexity the probability of unseen risks and the necessity of proper risk analysis increases. In the artifact proposed, we study the economic dependency (e.g. material or financial flow) between firms by means of BC calculation for the identification of risky firms in SCNs. We thereby assume that our artifact could be a module of standard ERP systems that use existing communication links to suppliers and customers. An alternative implementation could use existing blockchain technology.
- Preservation of Privacy: One of the main concerns of firms in a SCN is their strategic position in the network, so they avoid to risk their competitive advantage in order to identify their own risks. Our artifact keeps the network's structure mostly unknown to the firms within the network. The artifact prevents data leakage or reconstruction of

information to ensure the firms' willingness for information sharing. In order to meet this objective, we base our approach on SMC algorithms in a semi-honest environment as outlined in the latter. Our modeling focus is on providing a privacy preserving artifact, whereas we omit the analysis and improvement of computational complexity.

Considering the guidelines by Hevner et al. [31] and Gregor and Hevner [32] for the conduction of design science research, the remainder of this papers is organized as follows: The first section covers a brief review on essential literature. It also includes specifying the problem's context and the relevance of the problem for SCNs. Subsequently, we discuss the modeling procedure and requirements that must be met for solving the problem. The fourth section illustrates the developed artifact. The section is followed by the evaluation of the artifact by means of testing and descriptive methods. The paper ends with a summary and an outlook on further research.

2. Literature Review

2.1 Supply Chain Networks

"Supply chains are interlinked networks of suppliers, manufacturers, distributors and customers that provide a product or service to customers" [33]. Current trends, like e-commerce, e-logistics, and e-business, increase the complexity of supply chains. Furthermore, the importance of staying competitive in the market gives supply chain management a higher importance [34]. The SCN in a global economy consists of a large number of interdependent networks. This interdependency is very susceptible to external effects and defaults [35]. The risk type in SCNs can be specific disruption, general disruption, cost shock (e.g. exchange rates), product safety, commoditization, and shift in tastes [30]. Weather, terrorism, firms manufacturing failures, or financial crises can cause a default in the supply chain [36]. Risks in SCNs can lead to various types of losses such as financial loss, performance loss, physical loss, psychological loss, social loss and time loss [37]. Since the disruptions in SCN in extreme cases may lead to the bankruptcy of the SCN's firms, it is important for the firms to manage these risks and minimize the possible losses. A study by Gyorey et al. [38] states that 67% of firms are not ready for geopolitical instability challenges. In the management of SCNs, one of the main tasks is risk management. The risk management process consists of risk identification and assessment, decision and implementation of risk management actions, and risk monitoring [21]. Bellamy and Basole [39] classified the themes in SCNs analysis as system architecture (network structure), system behavior, and system policy and control. Among these categories, system architecture analysis methods focus on structural investigation of SCNs, relationship of firms, and the importance of the relationship. Considering social networks, structural investigations based on network analysis methods are well-established. In the field of SCNs they are relatively new but evolving [15,17,40]. These methods focus on network components' connections and patterns, and implication of these connections for the whole network [18,20]. Among various measures on structural analysis of SCN, as it has been mentioned earlier, the BC can be a suitable indicator to identify the structural risks of a SCN [15] and it is our choice in this paper.

2.2 Privacy Concerns in Supply Chain Networks

On the one hand knowing the structure of a network is a prerequisite of calculating the BC (as outlined earlier) and on the other hand in a SCN, the competitive advantage of network firms is relying on the privacy of their contacts and network relations they have [35]. Solutions to these data privacy concerns of firms can be:

- A Trusted Third Party: If the firms trust a third party, it is easy to solve the problem by sharing their information with this trusted third party and letting it calculate the results. For instance, Brandes' algorithm for the BC [23], works based on the idea of having a third party who collects the information and calculates the indices and returns the result. In practice such a party that all network's firms trust might be difficult to find and firms might have concerns about this third party revealing the information.
- SMC Algorithms: These cryptography algorithms enable different firms in the network to share their information privately and calculate the result jointly. The main advantage of these algorithms is that the individual's input stays mostly private.

SMC first was addressed by Yao [41]. Yao's algorithm is answering the question of SMC for two parties. This algorithm is a solution to the Millionaires' problem. The problem is that two millionaires want to know which of them is richer but they do not want to share the real amount of their wealth. Yao's [41] algorithm provides a solution that lets them privately encrypt their input, share it, and jointly calculate the result. The main advantage is that their input stays private. SMC algorithms today enable us to do secure addition, multiplication, and comparison [25,42–44].

SMC algorithms are used in various fields of science. For instance they are used for secure auctions [45]. They are also used for sharing financial risk exposures [46] with the focus on necessity of process and methods secrecy in financial industry. SMC algorithms are also applied for sustainable benchmarking in clouds without disclosing the individual's confidential information [47].

"SecureSCM", secure collaborative supply chain management, the European research project [48], is an example of the application of SMC algorithms in the field of SCNs. The project enabled privacy preserving online collaboration among various firms in a SCN. The focus was on providing the possibility to better reaction on possible capacity concerns or short notices. The collaboration of the firms with the application of SMC algorithms results in better production planning in the SCN. However, they did not study SCN's risks and focused on cost minimization.

In this paper, SMC algorithms are our choice for the privacy preserving calculation of the result. To apply these algorithms, we develop an artifact that enables calculation of the result based on private shares of the firms. SMC algorithms have a high acceptance and are widely used in the field of cryptography since the 1980's [45,49–52] as their security has been addressed comprehensively, as well.

2.3 Network Centrality Measures

To calculate the BC, we model the SCN as a graph $G(V, E)$. Each firm v in the SCN is represented by a vertex $v \in V$. An economic dependency (e.g. material or financial flow) between firms $u, v \in V$ is represented by an edge $(u, v) \in E$ between these firms. In this case, we name u and v adjacent or neighbors. Since an economic dependency is undirected, in this paper graphs are undirected. Moreover the graphs are connected, as connected firms are forming a SCN. The BC is a centrality index based on the number of shortest paths and the frequency in which a vertex is appearing on shortest paths between two other vertices. A shortest path is a path between two vertices such that the sum of the weights of its constituent edges is minimized (as outlined in Section 3). The BC describes how other vertices potentially can influence the interaction between two non-neighboring vertices [18,20]. The BC for vertex v is calculated as follows [18]:

$$BC(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{1}$$

In Equation (1), $\sigma_{st}(v) \in \mathbb{N}_0$ is the number of shortest paths between source vertex s and target vertex t, which pass through vertex v, and σ_{st} is the number of shortest paths between source vertex s and target vertex t.

The main aspect of the BC algorithms [23,53,54] is finding the shortest paths. Based on categorization of Cormen et al. [55] on shortest paths algorithms we classify existing BC algorithms as follows:

- Algorithms based on single-source shortest paths: Brandes' algorithm [23] is a widely used one among them. Brandes [23] applies single source shortest paths algorithms (breadth-first [56]) search for unweighted and Dijkstra's algorithm for weighted graphs [55,57] to calculate the BC.
- Algorithms based on all-pairs shortest paths: The method developed by [58] adopted modification of algorithms like the Floyd-Warshall [55,59,60] to enable parallelism and space-efficiency in calculation of the BC.

Both categories of algorithms need the network topology as input and a stack to store information. For privacy concerns we strive to avoid a central stack for information. Having a central stack implies that there is a central player who owns this stack. This player can infer information, from the communication of the players via this stack or from the large amounts of available data (although the information is encrypted) in the stack. This can be a risk for privacy concerns of the firms in the SCN.

In this paper, inspired by the Floyd-Warshall [55,59,60] algorithm as well as backtracking search [61] to identify shortest paths, we develop an artifact which does not need a central stack, stores information decentrally, and does not need the network's topology as input.

3. MODELING PROCEDURE, ASSUMPTIONS, AND REQUIREMENTS

The first part of this section focuses on the modelling procedure and assumptions of our artifact. In this part, before we focus on privacy concerns and information that each firm has, we define the general terms and construct of our artifact. The second part includes the more specific information on the firm's privacy preservation and requirements.

We label each firm and its representing vertex with a unique number $1,2,\dots,|V|$. The numbers are randomly assigned to each firm and represent the row number for the player in the graph's weight matrix. The relation between the identity of a firm and its number is only known to the firm itself and to the neighboring firms. From now on, we name a firm and its representing vertex as a "player" when we refer the firm's row number and not the true identity of the firm.

In the following, we illustrate an exemplary SCN (Figure 1). The SCN is chosen simple to make the visualization easier and the example more comprehensible. The SCN consists of 7 players. Each player is represented by its own unique number. The set of vertices (players) is: $V = \{1,2,3,4,5,6,7\}$.

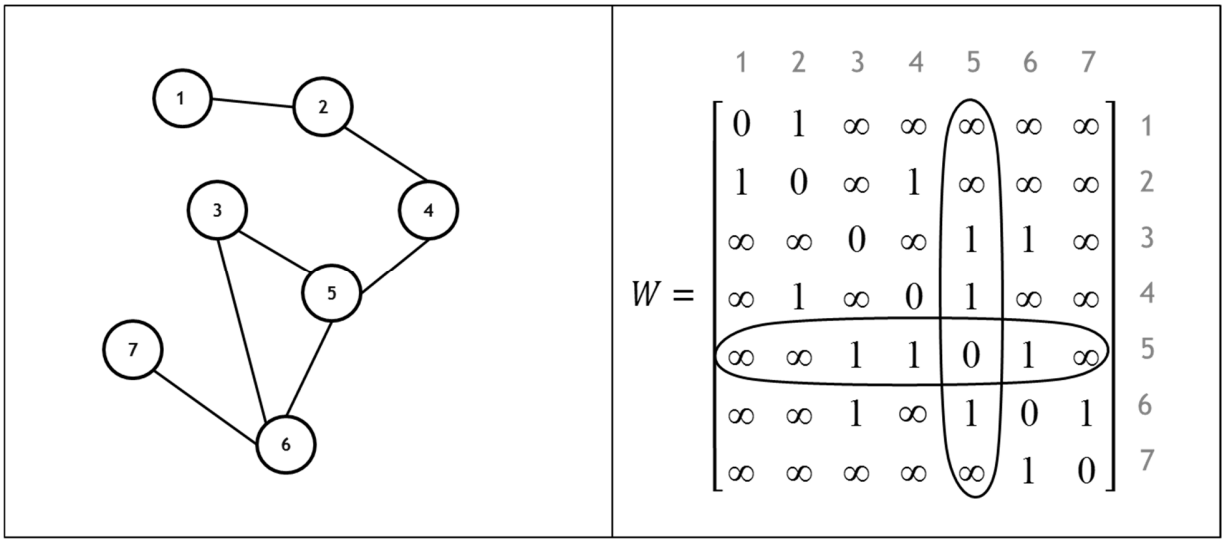

Figure 1 Exemplary network [27]

For reasons of simplicity, the following assumptions are the basis for the development of our artifact.

Assumption 1. *The firms are semi-honest (honest-but-curious).*

Semi-honest adversaries are following the protocol, but they might try to gather information and draw conclusions from the messages they receive. Our artifact's construction preserves privacy assuming the firms are semi-honest. Moreover, related works on SMC algorithms are also based on a semi-honest model [62–65].

Assumption 2. *The connections in the SCN are equally weighted.*

In general, our artifact is applicable for graphs with $w_{uv} \in \mathbb{R}$. However, Kim et al. [15] did their analysis on the BC, assuming equal weight connections. Their focus is on links between firms and the number of firms that are engaged in transferring information or material. Therefore, without loss of generality, in this paper we do not focus on the determination of the intensity of connections and its analysis and we treat the connections as equally weighted and leave the topic of connections' intensity subject to further research. The weight of the edge $(u, v) \in E$ with arbitrary $u, v \in V$ is then defined by

$$w_{uv} = \begin{cases} 0 & if\ u = v, \\ 1 & if\ u \neq v\ and\ (u, v) \in E, \\ \infty & if\ u \neq v\ and\ (u, v) \notin E. \end{cases} \quad (2)$$

The $n \times n$ matrix $W = (w_{uv})$ contains all weights of edges in the graph $\forall\ u, v\ \in V$ [55]. The (symmetric) matrix W in Figure 1 represents the weight matrix of our exemplary SCN.

The sequence of vertices that are forming the path from a source vertex $s \in V$ to a target vertex $t \in$ V is represented by $path = \langle v_0, v_1, \ldots, v_k \rangle$. In this we assume that $v_0 = s$, $v_k = t$, and $(v_{i-1}, v_i) \in E$ for $i = 1\ to\ k$. The length of the path is the sum of the weights of its forming edges. Based on Equation (2) the weight of an edge is 1 therefore, if k vertices are forming a path, there are $k - 1$ edges on this path and $w(path) = k - 1$. We define the length of a shortest path, labeled as distance between s and t, as

$$d_{st} = \min\{w(path) : v_s \overset{path}{\rightsquigarrow} v_t\} \quad (3)$$

The $n \times n$ matrix $D = (d_{st})$ contains the distances $\forall\ s, t\ \in V$. By our definition, if s and t are adjacent then $d_{st} = 1$. To find a shortest path from a source vertex s to the target vertex t, the existing distance and the distance of all alternative paths via intermediate vertices $\forall v \in V, v \neq s, t$ are compared (Equation (4)) and we choose the path with the minimum length.

$$\min(d_{st}, d_{sv} + d_{vt}) \quad (4)$$

In this part we represent the above mentioned figures with particular details which include privacy preserving concerns and information availability for the players.

In our artifact we restricted the information availability of the players mostly up to their neighbors. Therefore, although the set V is known to every player in the network, but the relation between the players' unique numbers and their true identities is in only known to neighboring players. Furthermore the network's structure as illustrated in the Figure 1 is not known to the players. Consequently W is unknown to the players. Each player p has access to the $p - th$ row/column (since the matrix is symmetric) of the weight matrix W. The accessible information for player 5, is the 5-th row of the matrix, as marked in the Figure 1. Moreover the distance matrix D is unknown to the players. Although, each player p has access to the p-th row of the matrix D.

For our artifact we state the following requirements:

Requirement 1. *The artifact should keep the SCN topology as private as possible.*

Requirement 1 is an extension to conditions of SMC on satisfying privacy [44]. In our case it is not allowed that more information than the final result (BC) is shared. More specifically, we prohibit the sharing of the following information that can be used for reconstructing the SCN topology or interfering the real identity of the firms.

- The length of the shortest paths, to prevent firms from knowing the positioning of the players in the network.
- The number of the shortest paths between a given source and target player in the network, to prevent firms from knowing which alternatives for trading players have in the network.
- The number which shows how often a player is appearing on the shortest paths between a given source and target player, to prevent firms from knowing accessibility and connections to other firms.

Requirement 2. *The artifact should keep the identities of non-neighboring players private.*

In a large SCN, due to members' variety and multiplicity in the SCN, a firm is not able to identify other firms in the network. Concluding the identity of a player via execution of the artifact can provide the possibility of reconstructing a part of the network's topology. Therefore, the artifact should not enable a firm to infer the real identity of non-neighboring firm.

4. Artifact Development

We choose an object oriented approach to design the artifact. To model the structure and behavior of the players in our artifact we model the class Player. We represent each player by an object of class Player running on a distributed system. Each player executes the methods on its own system and delivers the result. In our artifact we assume there is an initializing and synchronizing agent (ISA) (one of the SCN's firms or an organization) who initializes, coordinates, and synchronizes the executions. The ISA does not have the possibility to access the private information of the players or monitor the communication between the players.

Figure 2 presents class Player. For reasons of simplicity, in the following we assume the players' object references equal to their respective $rowNumber$ during the calculations. $rowNumber$ is the unique number assigned to each player in the network. $rowNumber = p$ implies the player is pointing the p-th row/column the weight matrix W.

We assume p is the number of the current object of the Player class. Table 1 provides the description of the attributes of the Player class. Table 2 provides an overview and description of the commonly used variables in the methods. Table 2 provides the description of the methods of the Player class.

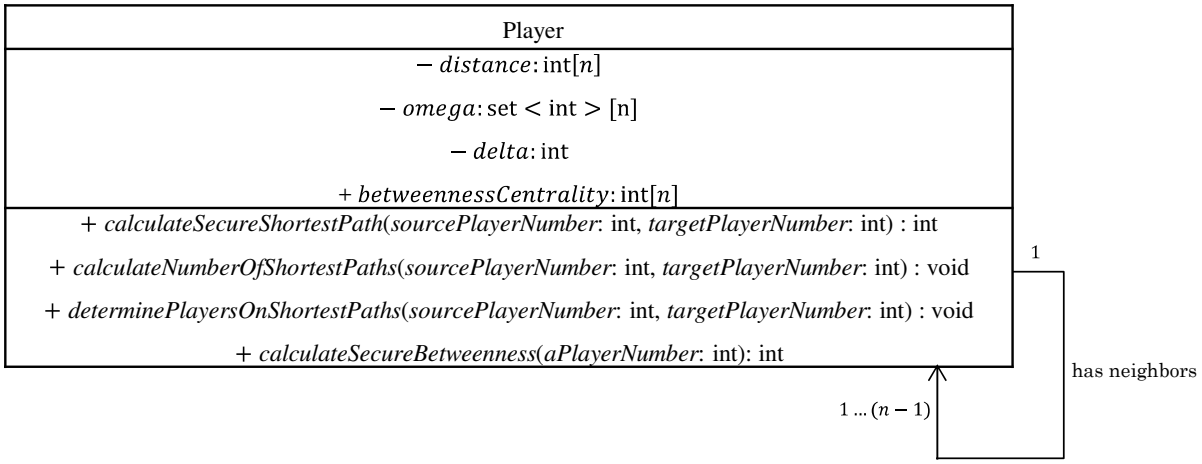

Figure 2 Visualization of the class Player in UML-oriented Notation [27]

Table 1 Description of the attributes of the class Player [27]

Attribute	**Mathematical Variable**	**Description**
$distance: \text{int}[n]$	$D = (d_t)$ $\forall\, t \in V$	Denotes a vector of the distances of player p to each target player t in the network. The distances are unknown at the beginning of the execution. Each member of this list is the output of the method *calculateSecureShortestPath()* for a given target player.
$omega: \text{Set} < \text{int} > [n]$	$\Omega = (\Omega_t)$ $\forall\, t \in V$	Denotes a vector which contains the set of neighboring players of player p that are connecting the player with the shortest paths to the target player t The method *calculateSecureShortestPath()* sets the values of this set.
$delta: \text{int}$	δ	Denotes a random generated number of the player. We use it to modify the distance value to preserve privacy. Each player generates δ before participating in the execution of methods. For each player, this number stays constant during the execution of the artifact. It assures an identical response of the player to all calculation requests.
$betweennessCentrality: \text{int}[n]$	$BC = (bc_v)$	Denotes a vector which is filled with the BC of all players in the network. Each member of this list is the output of the

Attribute	Mathematical Variable	Description
	$\forall\, v \in V$	*calculateSecureBetweenness()* method for each given player v.
$busy$: bool$[n][n]$	$BZ = (bz_{st})$ $\forall\, s,t \in V$	Denotes an $n \times n$ matrix of flags (true/false). This flag serves implementation purposes and especially access management of the players (as described in Table 4, Line 1-4).

Table 2 Description of the methods of class Player [27]

Method	Description
Name *calculateSecureShortestPath* **Input** sourcePlayerNumber: int targetPlayerNumber: int **Output** distance: int	The method recursively identifies the shortest paths from the given source player to the given target player. It returns the encrypted value of the distance and keeps other variables local. If the target player is not the current player, the method calls itself at all neighboring players to determine their distances to the target. The method compares the delivered results of the neighboring players and chooses the path via the neighboring player/s which has/have the minimum distance value. For privacy preserving purposes the comparisons in this method are based on Yao's secure comparison protocol [41]. The method also identifies the neighboring players who are forming the shortest paths and fills the set Ω.
Name *calculateNumberOfShortestPaths* **Input** sourcePlayerNumber: int targetPlayerNumber: int **Output** void	The method recursively calculates the *number* of shortest paths between given unique numbers of source player s and given target player t via players forming the shortest paths. If t is not a neighboring player of s, the method calls itself at all neighboring players forming the shortest paths between s and t. The method determines the number of shortest paths which passes through current player p by means of the size of the set Ω_t. The method saves the results of the calculation in an intermediate storage and later uses it to participate in *calculateSecureBetweenness* method.
Name	The method recursively determines how often players are appearing on the shortest paths from source player s to target player t via current player p. If

Method	Description
determinePlayersOnShortestPaths **Input** sourcePlayerNumber: int targetPlayerNumber: int **Output** void	t is not a neighboring player of s, the method calls itself for all neighboring players which are forming the shortest paths between s and t. At each recursion the members of the set Ω_t determine the players which are on the shortest paths through player p. The method determines the players which are on the shortest paths through current player p by means of the members of the set Ω_t. The method saves the results of the calculation in an intermediate storage and later uses it to participate in *calculateSecureBetweenness* method.
Name *calculateSecureBetweenness* **Input** sourcePlayerNumber: int targetPlayerNumber: int **Output** BC(v): int	This method calculates the $\mathrm{BC}(v)$ for the given player in the network. It is based on SMC algorithms and requires information exchange among the players in the network. The method performs all arithmetic based on secure protocols of Cramer et al. [44]. These protocols for SMC are an extension of Shamir's algorithm [42] and providing us the possibility to calculate the BC preserving the privacy concerns. Furthermore, the method applies the distributive property of binary operations to calculate the result of Equation (1). This provides us the possibility that private shares of players stay private.

For privacy preserving concerns, in methods *calculateSecureShortestPath(), calculateNumberOfShortestPaths(),* and *determinePlayersOnShortestPaths()* players only communicate via their neighboring players. Each object routes its encrypted messages through neighboring players in the network. The methods *calculateNumberOfShortestPaths*() and *determinePlayersOnShortestPaths*() calculate values of σ_{st} and $\sigma_{st}(v)$ decentrally. Each player has a portion of these values from its own perspective. We denote the portion of information which player p has by σ_{st}^{p}, and $\sigma_{st}^{p}(v)$. The final values of σ_{st} and $\sigma_{st}(v)$ are the sum of the decentrally calculated values of all players as follows.

$$\sigma_{st} = \sum_{p \in V} \sigma_{st}^{p}, \qquad \sigma_{st}(v) = \sum_{p \in V} \sigma_{st}^{p}(v). \tag{5}$$

The method *calculateSecureBetweenness*() uses the decentral values (σ_{st}^{p}, and $\sigma_{st}^{p}(v)$) to calculate the betweenness centrality, and applies SMC algorithms to preserve privacy.

Table 3 elaborates on the sequences of our artifact. Steps 1 to 5 (initialization) and 9 (synchronization) of Table 3 are not in the focus of this paper and also do not influence the artifact's construction. Therefore, these steps are not documented in this paper. Furthermore, we provide a brief description of all methods which are listed in Table 2.

Table 3 The artifact's sequences [27]

Step	Executor	Description
Initialization		
1	ISA	Identifies the number of players, n, in the network.
2	ISA	Assigns each participating company a $rowNumber$ (without knowing the real identities of the firms).
3	ISA	Shares the number of players, n, with all players in the network and notifies the players to initialize.
4	Player	Each player initializes a new object of class Player and informs ISA.
5	ISA	Notifies all players that the players' objects exist and they are available to execute the methods.
Decentral calculation of the shortest paths and path forming players		
6	Player	Each player executes the *calculateSecureShortestPath*() method for itself as the source player and all given targets in the network.
7	Player	Each player executes the *calculateNumberOfShortestPaths*() method to decentrally set the values of σ_{st} for each given target t.
8	Player	Each player executes the *determinePlayersOnShortestPaths*() method to decentrally calculate the values of $\sigma_{st}(v)$ for itself as source player s and each given target t. By termination of the method for all given targets, the player informs ISA.
Synchronization		
9	ISA	ISA informs every player in the network that the *determinePlayersOnShortestPaths*() is terminated when it receives the notification of termination from all players. This implies that the variables to calculate the BC are available.
Calculation of the BC		

Step	Executor	Description
10	ISA	ISA coordinates players for execution of the *calculateSecureBetweenness*() method. With termination of the method for all players in the network, all firms have their own BCs as well as the BC of all players in the network.

In the following, we provide the pseudocodes and a detailed description of the methods of our artifact. The artifact's methods use integer variables to reference players similar to the mathematical variable e.g. for inputs and outputs. The declared references are the source player s ($sourcePlayerNumber$), the target player t ($targetPlayerNumber$), the current player p ($currentPlayerNumber$) i.e. the player currently calling a method, a neighboring player a ($neighboringPlayerNumber$), and some given player v ($aPlayerNumber$). We present a summary on these variables in the appendix

In Figure 3 we provide the pseudocode of *calculateSecureShortestPath*() method. This method requires an additional variable td for calculation purposes. It denotes a temporary variable saving the distances during calculation of the shortest paths. This variable ensures data consistency.

```
METHOD 1.  calculateSecureShortestPath
Input: sourcePlayerNumber s, targetPlayerNumber t.
Output: d_t.
1   if bz_st = true then
2   {
3           return ∞
4   }
5   if t = p then
6   {
7           return δ
8   }
9   if d_t ≠ ∞ then
10  {
11          return d_t
12  }
13  else
14  {
15          bz_st = true
16          td = w_pt
17          for a = 1 to n do
18          {
19                  if w_pa = 1 then
20                  {
21                          if smin(a.calculateSecureShortestPath(s,t), td − w_a) then
22                          {
23                                  if s = p then
24                                  {
25                                          Ω_t ← {a}
26                                  }
27                                  td ← a.calculateSecureShortestPath(s,t) + w_a
28                          }
29                          else
30                          {
31                                  if ¬smin(td − w_a, a.calculateSecureShortestPath(s,t)) then
32                                  {
33                                          if s = p then
34                                          {
35                                                  Ω_t ← Ω_t ∪ {a}
36                                          }
37                                  }
38                          }
39                  }
40          }
41          bz_st ← false
42          return td
43          }
```

Figure 3 Pseudocode of the method calculateSecureShortestPath

Table 4 provides a detailed description of the *calculateSecureShortestPath*() method.

Table 4 Description of the calculateSecureShortestPath() method

Line	Description
1-4	The *calculateSecureShortestPath*() is a recursive method, which sequentially routes the requests of the calculation of shortest paths via the neighboring players. Therefore, each player should not receive a duplicate request for the calculation of a specific path. However, a player may receive such a request, since the graph of the SCN is not necessarily acyclic. Once current player p routes the message of the calculation of a specific shortest path via a neighboring player, due to the possible graph cycles, after few message routings player p might receive its own message from a neighboring player causing endless loop. To avoid such conditions, the method uses a busy flag (bz_{st}). As long as player p is busy with the calculation of the shortest paths between players s and t, if it receives a message for the calculation of the same path, it implies that the message is its own message. Therefore, Line 1 identifies this message as a duplicate message. Furthermore, Line 3 prevents further calculations of the method and returns ∞. Returning ∞ ensures that the duplicate request has no influence on the result of the calculations, and the method terminates.
5-8	To calculate the BC (c.f. Equation (1)), it is important to know the number of the shortest paths between two players, and to know which players are forming the shortest paths. The *absolute numeric value of the length* of the shortest paths does not change the result of BC. Thus, for privacy preserving concerns, we can modify the *absolute numeric value* of the distances between players by adding an offset to the target players given that the number of shortest paths and their forming players remain intact. Still, we obtain the same results as without modification of the distances. In our method, each player uses its own private number $\delta \in \mathbb{N}$ (delta explained in Table 1) to modify the distance value (Line 7). Note that, this number must not be the players' unique number, because if it is so, the positioning of the players might be disclosed. Please note, since the communication is only via neighboring players, this private number is only known to the player and its neighboring players. Players use the private number δ only if they are the target player t. This assures a consistent modification of the distances to a specific target player t and the comparability of the results for the source player s. The following scenario elaborates the importance of using δ to modify the value of distance. In our exemplary network (Figure 1), if the method does not modify the value of distances, and player 6 shares 1 as its distance to players 7 ($d_{67} = 1$), player 5 (as a neighboring player of player 6) infers that players 6 and 7 are adjacent. But if the method uses a modified value of the distance (we define, δ for player 7 be 70), player 6 shares 71 as its distance to players 7 ($d_{67} = 71$). This modification hinders unwanted information sharing in terms of inferring the positioning of players in the network.

Line	Description
	To find the shortest paths, we must be able to compare the distances of the paths. Although the modified distance values (Line 7) eliminate many sorts of privacy concerns, yet there is a chance to reconstruct parts of the network structure by comparing the modified values. For instance in Figure 1 we set δ for player 7 to 70. The distance of player 3 to 7 via player 5 is $d_{37} = 73$ and the distance of player 3 to 7 via player 6 is $d_{37} = 72$. Based on this information, player 3 reveals that players 5 and 6 are adjacent. Therefore, in addition to modifying the shortest path we apply a privacy preserving method to compare the shortest paths (See Line 17-40).
9-12	If the player already calculated the distance to target player t, then it returns this calculated value of distance. This part increases the efficiency of the method by preventing recalculation of the shortest paths, which are already calculated.
13-43	If the current player p receives a request for the calculation of a specific path for the first time and is not the target player, this part of the method (Line 17-40) recursively calculates the shortest paths between source player s and target player t. To avoid data inconsistency during the execution of various instances of the method, Line 16 sets the temporary distance variable td_{st} to w_{pt} which is the initial distance value of the current player to the target. The method does not use its distance attribute (d_t) for calculations, because the value of d_t, may change during the calculation of a specific shortest path, leading to inconsistency of the result. The following example elaborates the necessity of the temporary distance variable. We assume player 5 is executing *calculateSecureShortestPath*(1,7) and it is the first request to player 5 for calculating the path to player 7. Players 5 and 7 are not adjacent and the initial value of the distance is $d_7 = w_{57} = \infty$. In the meantime, player 5 receives the request for calculation of the path from source player 2 to target player 7 (*calculateSecureShortestPath*(2,7)). If the execution of this request ends faster than *calculateSecureShortestPath* (1,7), player 5 updates the distance to player 7 (d_7) to 72. Consequently, the value of d_7 for the player 5 varies during the execution of *calculateSecureShortestPath*(1,7).This leads to inconsistent values of the distance for the comparisons within the execution of the method. Using td, the method prevents this sort of inconsistencies. When player p starts the calculation of the path between source player s and target player t via its neighboring players, Line 15 sets the bz_{st} to true. When player p finishes calculating the shortest paths between players s and t, Line 41 sets bz_{st} to false. It allows the player to respond to the messages which are not originating from itself.

Line	Description
	At Line 42 the method returns the value of distance (td), which is the distance of player p to the target player t (d_t).
17-40	Current player p routes the message of the calculation of the path via all neighboring players to calculate the result recursively. For this purpose Line 17 goes through each player 1 to n, where n is the number of players in the network. Furthermore, the method identifies the neighboring players and only routes the request for calculating the shortest path via them.
19-39	Player p identifies its neighboring players at Line 19. For this purpose it considers a given player a as a neighboring player when the distance of the current player to the player is equal to one ($w_{pa} = 1$ (Equation (2))). The player routes the request of calculation only via its neighboring players, since for privacy preserving concerns we limit direct communication of players and only allow communication via neighboring players. Please note that in this paper we assume the connections in the SCN are equally weighted. In a weighted graph, another mechanism to identify the neighboring players will be necessary. Furthermore, this part of the method determines a new shortest path (Line 21-28), or the additional shortest paths (Line 29-38).
21-28	This part of the method determines if the path via the neighboring player is a new shortest path. We define, *smin*(m, n) as a function which performs the comparison of given input parameters m and n based on Yao's secure comparison algorithm as $$smin(m,n) = \begin{cases} \text{true} & \text{if } n < m, \\ \text{false} & \text{otherwise} \end{cases}$$ which keeps the input parameters of the players n and m private. In method *calculateSecureShortestPath()* we are interested in finding the result of $smin(w_a + d_t^{(a)}, d_t)$. The values of d_t and w_a are known to player p. The distance of the neighboring player a to target player t, ($d_t^{(a)}$) is known to player a. To use $smin()$, and keep the input variables of each player private, we do the comparison as $smin(d_t^{(a)}, d_t - w_a)$ which uses the input of the current player and the neighboring player separately. The value of $d_t^{(a)}$ is not known for the current player, therefore it routes the requests to its neighboring player to participate in the calculation of the $smin$ by $smin(a.calculateSecureShortestPath(), td - w_a)$. Please note, as mentioned at Line 16 current player uses a temporary distance value td instead of d_t during the calculation.

Line	Description
	Player a does not share the distance value, and takes part with its encrypted input in the secure comparison of the distances. If the result of smin($d_t^{(a)}, d_t - w_a$) is true, it implies that the alternative path via a is shorter than the existing path(s), so this path is a shortest path in this iteration. In this case current player, at Line 27, assigns the calculated value of distance to its temporary variable of distance td. For privacy preserving concerns, we share as little information as possible. Consequently, if the alternative path via player a is not shorter than the existing one(s), current player will not find it out.
23-26	Since the method is recursive, it is important to prevent assignment of values during the execution and before the source player which initiated the request receives the final result. Line 23 examines if p is source player s, which implies the initiating player received its own request, and then allows the method to update Ω_t. By finding a new shortest path (at Line 21), the previously found path(s) and the players which are forming these paths are not relevant anymore. Player a is the neighboring player, which connects player p with the shortest path to the target. Therefore, the method updates Ω_t, and sets player a as its only member.
29-38	If the path via the neighboring player is not shorter than previously found shortest path(s), this part of the method determines if the path via this neighboring player is an additional shortest path.
31-37	This part of the method aims to determine if the path via the neighboring player is an additional shortest path. If the following equation is true, it implies that the alternative path via player a, and the already calculated path are equal. $$\neg smin(m,n) \wedge \neg smin(n,m) = true$$ In our case we evaluate the following expression: $\neg smin(d_t^{(a)}, d_t - w_a) \wedge \neg smin(d_t - w_a, d_t^{(a)}) = true$. In the case of $\neg smin(d_t^{(a)}, d_t - w_a) = true$, we only need to examine $\neg smin(d_t - w_a, d_t^{(a)}) =$ *true*. Line 31 of the methods performs this comparison. For privacy preserving concerns (as already elaborated at Line 21-28) player a does not share the distance value, but only takes part with its encrypted input for secure multiparty calculation of $smin(td - w_a, a.calculateSecureShortestPath())$. It should be noted that for the comparison of the shortest path distances (Line 31) the current player does not know $d_t^{(a)}$, and therefore routes the request to the neighboring player a by

Line	Description
	a.calculateSecureShortestPath(s,t). Please note, the neighboring player a, already calculated this path (as Line 21) and therefore immediately returns this value.
33-36	Since the method is recursive, it is important to prevent the assignment of values before the source player, which initiated the request receives the final result. Line 33 examines if p is source player s and then allows the method to update Ω_t. Finding an additional shortest path implies that player a is connecting player p with the shortest path to target player t. Therefore, Line 35 adds player a to Ω_t.

For reasons of simplicity we provide the sequence diagram of the method for a specific path. Figure 4 provides the *calculateSecureShortestPath*(5,7) from player 5's perspective for our exemplary network (Figure 1). We assumed δ for player 7 is 70.

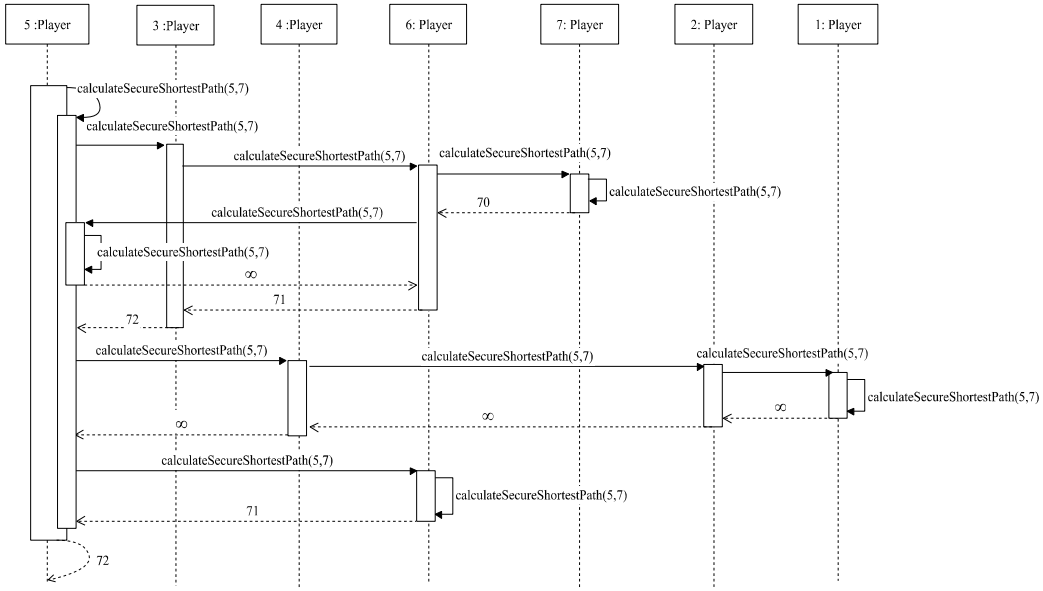

Figure 4 Sequence diagram for calculateSecureShortestPath(5,7) from player 5's perspective [27]

In the following, Figure 5 presents the pseudocode of *calculateNumberOfShortestPaths*() method.

```
METHOD 2.    calculateNumberOfShortestPaths
Input: sourcePayerNumber s, targetPlayerNumber t.
1    if s = p then
2    {
3        if w_t = 1 then
4        {
5            σ_st ← 1
6        }
7        else
8        {
9            σ_st ← |Ω_t|
10       }
11   }
12   else
13   {
14       σ_st ← max(|Ω_t| − 1, 0)
15   }
16   for each ω ∈ Ω_t do
17   {
18       ω.calculateNumberOfShortestPaths(s, t)
19   }
```

Figure 5 Pseudocode of the method calculateNumberOfShortestPaths

Table 5 provides a detailed description of *calculateNumberOfShortestPaths*() method.

Table 5 Description of the calculateNumberOfShortestPaths () method

Line	Description
1-11	This part of the method sets the number of the shortest paths (σ_{st}) when the current player is the source player s.
3-6	This part of the method sets the number of the shortest paths (σ_{st}) when the target player is a neighboring player of source player s. If target player t is a neighboring player of source player s (Line 3), Line 5 sets the number of the shortest paths to one ($\sigma_{st} = 1$) because there is only one shortest path between two neighboring players in an unweighted graph.
7-10	This part of the method sets the number of the shortest paths (σ_{st}) when the target player is not a neighboring player of source player s. If target player t is not a neighboring player of source player s, the size of set Ω_t (that includes all neighboring players which connect current player p as source to the target) is the number of shortest paths between player p and the target t (σ_{st}). As already mentioned, the player has a portion of this value from its own perspective. The final value of σ_{st} is the sum of the decentrally calculated values of all players as shown in Equation 5.
12-15	This part of the method sets the number of the shortest paths (σ_{st}) when the current player is not the source player s. This player received the request of *calculateNumberOfShortestPaths(s,t)* because it is one of the players which is forming the shortest path between source and target player. Already one of the shortest paths on which the player lies, is considered by the source player s. Consequently, we need to consider the additionally identified shortest paths via this player. If there is an additional path via this player to the source, Line 14 sets the value of σ_{st} to $\lvert\Omega_t\rvert - 1$, otherwise set it to zero. We decrease the value of $\lvert\Omega_t\rvert$ by one, to prevent double consideration of the already considered path.
16-19	To consider additional shortest paths which might be identified by the players which are forming the shortest paths between the source and target player, the method recursively routes the message for calculating the number of shortest paths via the neighboring players which are forming the shortest paths (Line 18). The method identifies this neighboring player by Line 16, when $\omega \in \Omega_t$.

The *calculateNumberOfShortestPaths*() method identifies the number of the shortest paths from the source player and recursively identifies additional shortest paths via the players who are forming the shortest path(s). The following

example elaborates an exemplary scenario of the method's execution. For instance player 5 executes the *calculateNumberOfShortestPaths*(5,7) and identifies $\sigma_{57}^{(5)} = 1$. Since player 7 is not a neighboring player of player 5, and player 6 is in Ω_7 player the method calls itself from player 6. Player 6 does not identify any additional path (since player 6's $\Omega_7 = 0$) therefore, it sets $\sigma_{57}^{(6)} = 0$. At this point the method terminates while player 7 (the target) is a neighboring player of player 6.

Figure 6 provides the pseudocode of *determinePlayersOnShortestPaths*() method.

METHOD 3. determinePlayersOnShortestPaths

Input: sourcePlayerNumber *s*, targetPlayerNumber *t*.

```
1   if |Ω_t| > 1 and s ≠ p then
2   {
3       σ_st(p) ← max(|Ω_t| − 1, 0)
4   }
5   for each ω ∈ Ω_t do
6   {
7       σ_st(ω) ← 1
8       ω.calculatePlayersOnShortestPaths(s, t)
9   }
```

Figure 6 Pseudocode of the method determinePlayersOnShortestPaths

Table 6 provides a detailed description of *determinePlayersOnShortestPaths*() method.

Table 6 Description of determinePlayersOnShortestPaths() method

Line	Description
1-4	This part of the method sets the values of the frequency of the appearance of a player on a shortest path for itself $\sigma_{st}(p)$. When current player p is not source player s and the number of shortest paths (size of set Ω_t) from the current player to the target is greater that one, Line 3 sets the value of $\sigma_{st}(p)$ to $\lvert\Omega_t\rvert - 1$. The player is already considered on the shortest paths by the neighboring player which called it. Therefore, to prevent double consideration of the player we decrease the value of $\lvert\Omega_t\rvert$ by one. As already mentioned, the player has a portion of this value from its own perspective. The final value of $\sigma_{st}(p)$ is the sum of the decentrally calculated values of all players as shown in Equation 5.
5-9	This part of the method sets the values of the frequency of the appearance of a the neighboring players that form the shortest path on the shortest path between players s and t ($\sigma_{st}(\omega)$) and routes the message via the neighboring players forming the shortest paths. Line 7 sets the value of $\sigma_{st}(\omega)$ for the neighboring player ω to one because the player ω is on the shortest path from s to t. Player p can only update the values of $\sigma_{st}(\omega)$ for its neighboring players, but the frequency of appearance of a player on the shortest paths should be updated for all of the players on the shortest paths between source player s and target player t. The method calls itself to route the message via its neighboring player and update the values recursively.

The *determinePlayersOnShortestPaths*() method subsequently considers a player on the shortest paths between source player s and target player t when the player is in Ω_t of the current player. The following example elaborates an exemplary scenario of the method's execution. Moreover it reconsiders the current player (except the case where $s = p$) on the shortest paths when current player p has more than one shortest path to the target. For instance the *determinePlayersOnShortestPaths*(5,7), identifies $\sigma_{57}^{(5)}(6) = 1$ while player 6 is in player 5's Ω_7. Since player 7 is not a neighboring player of player 5, the method calls itself from its neighboring player (player 6). Player 6 is the neighboring player of the target (player 7) therefore, no further calculation takes place and the method terminates.

The *calculateSecureBetweenness*(v) method calculates the BC for player v based on SMC algorithms. In order to facilitate all-to-all communication, ISA coordinates the simultaneous exchange of information. To ensure that the real identities of the firms stay private in an all-to-all communication, existing tools for anonymization can be adapted.

The BC for player v based on Equation (1) is as follows:

$\mathrm{BC(v)} = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} = \frac{\sigma_{12}(v)}{\sigma_{12}} + \frac{\sigma_{13}(v)}{\sigma_{13}} + \frac{\sigma_{14}(v)}{\sigma_{14}} + \cdots + \frac{\sigma_{ij}(v)}{\sigma_{ij}}$, where $i = |V|$ and $j = |V| - 1$.

For the calculation of the BC we use SMC algorithms. Secure addition and secure multiplication algorithms will, however, reveal a party's input as inverse functions can easily be applied for only two input factors. To keep the input variables in arithmetic operations private, it is necessary that more than two players deliver input. In the above mentioned equation we address this problem. By division of two variables delivered by two players, even with the application of SMC algorithms, the end result reveals the input variables for the players. Therefore, by using a common denominator we solve the problem as follows:

$$\mathrm{BC}(v) = \frac{\sigma_{12}(v) \cdot (\sigma_{12} \cdot \sigma_{13} \cdot \ldots \cdot \sigma_{ij}) + \sigma_{13}(v) \cdot (\sigma_{12} \cdot \sigma_{13} \cdot \ldots \cdot \sigma_{ij}) + \cdots + \sigma_{ij}(v) \cdot (\sigma_{12} \cdot \sigma_{13} \cdot \ldots \cdot \sigma_{ij})}{\sigma_{12} \cdot \sigma_{13} \cdot \sigma_{14} \cdot \ldots \cdot \sigma_{ij}} \quad (6)$$

Furthermore, the values of σ_{st} and $\sigma_{st}(v)\ \forall s \neq v \neq t \in V$ are the results of Equation (5). For privacy preserving concerns, as addressed in Requirement 1, we do not calculate and share the final values of σ_{st} and $\sigma_{st}(v)$ in the network. Hence, we use the distributive property of arithmetic operations to distributedly consider the components of Equation (5) in Equation (6). Using the mentioned modification on the BC calculation's equation we provide the possibility to keep the private shares of the players private and calculate the BC. The implementation of the artifact with the application of SMC algorithms, anonymization methods, and necessary communication protocols are not covered in this paper.

5. Evaluation

This section provides the evaluation of our artifact. Concerning characteristics of our artifact, we chose the "testing" and "descriptive evaluation" methods based on [31] and [66]. We implemented a simplified prototype of the artifact. The prototype covers the methods of class Player. However, the prototype does not cover the implementation of SMC algorithms and assumes they are given. Moreover, the prototype models each player as a local thread, and it is not executed on a distributed system. Furthermore, a third person other than the authors manually evaluated the artifact with a structural walk through the code. In the following we cover general evaluation of completeness, termination, complexity, utility and privacy of the artifact. Furthermore, we illustrate the privacy evaluation based on an application example. It is important to note that we exemplarily analyze and present the privacy situation of player 5. This is so because it is the player which obtains most information by the application of the algorithm. Because of the acceptance and wide application of SMC algorithms we did not analyze their properties again but assume SMC algorithms to be complete and secure.

Completeness: To evaluate the artifact in terms of completeness we executed the prototype with various scenarios and evaluated the results. It proved that our approach creates complete results for each given network. Moreover, the structural walk through the code resulted the same.

Termination: By means of testing the prototype in various scenarios as well as structural walk through the code we validated that the artifact terminates.

Complexity: Analysis of our artifact pointed both the time complexity and the message complexity are polynomial in the maximum distance between the source and the target player, and number of network members. In our artifact we focused to achieve a privacy preserving method. To preserve privacy, it is necessary for the players to encrypt and exchange data more often compared to some widely used algorithms (e.g. [23]). Further improvements of computational complexity of the artifact is subject to further research.

Utility: Based on Gregor and Hevner [32] an artifact evaluation must address the utility of the artifact. Due to the complexity of implementation and evaluation of the artifact's utility in reality, in this paper we evaluated the utility of the artifact based a simplified prototype, and used an application example. Our artifact's characteristics based on [66] are: it is a novel method, which is open because it is possible to modify it, and is interesting because it addresses risk management and sustainability as one of the main concerns of the firms in SCNs.

Privacy: The privacy requirements of our artifact (Requirement 1 and 2) are addressed as follows.

- The application of Yao's [41] comparison algorithm and using the modified values for distances ensure that the distances of non-neighboring players remain unknown. Although in a small network, we illustrate in our application example, the distances might be inferable. However, in larger real-world SCN (which are in the focus of our research) players cannot infer the distance during the execution of the artifact.
- The number of the shortest paths, and the frequency of appearance of a player on the shortest path are saved decentrally, as mentioned in Equation (5). Therefore, the final values of σ_{st} and $\sigma_{\mathrm{st}}(v)$ are not available to the players and stay private.
- By restricting communication via neighboring players and application of anonymization methods, we addressed Requirement 2.

However, we will appreciate if other researchers challenge our artifact in terms of privacy. In specific cases players might infer information when they are called from neighboring players to execute the methods. However, the inferred information of the players are limited to the information from their perspective. For instance if the shortest path of a neighboring player to target t is via the current player it implies for the current player that the neighboring player and target t are not neighbors. Whereas it does not contain the information about the players which are forming the shortest paths and the number of shortest paths.

Furthermore, to illustrate the potential of our artifact to preserve privacy, we describe the artifact's outcome in a short example. Figure 7 provides the network structure (See Figure 1) from player 5's perspective before and after execution of the method. Based on the result of the BC calculation, players are prioritized and colored as shown in the figure. Player 5 has the highest BC. Player 4 is second. Players 6 and 2 are having the same BC and thus rank third place. The BC of players 1, 3 and 7 is zero, because they are not on any shortest path. This is a valuable information for all

network's members. For instance it implies that if player 5 faces any failure, the whole network's robustness might be at risk. The BC of the players is available for all players in the SCN.

Privacy related issues stem from disclosed information such as return values from method calls. An essential method is calculateSecureShortestPath(s,t). It operates decentrally and discloses portions of information. We therefore present all return values (if called) that are available for individual players. The pseudocode does not store the return values but a curios player might do so. That is why we provide an additional analysis if and if so what information might be inferred. Because the path from s to t has the same length as the path from t to s (adjusted by the difference of their delta-values), we only present all pairs with $\{(s,t) \mid s < t\}$. Table 7 presents the shortest path information for all 7 players in our example as illustrated by figure 1 after all calculateSecureShortestPath(s,t) calls terminated. For the illustration we assigned a random delta-value to each player. Before, we have already assigned 70 as the delta-value for player 7. In this example we continue to do so. Also, in Table 7 we present a delta distance value and an actual distance value. While the former represents the return value of s.calculateSecureShortestPath(s,t), the actual distance value denotes the real, non-distorted distance between s and t. The latter information shall not be revealed or inferable by any means as has been formulated by the requirements. By the results depicted in Table 7, it becomes clear that more central players as indicated by the BC score receive more method calls and thus more information.

Table 7 Shortest path information from the perspective of each player

Player number (p)	**1**	**2**	**3**	**4**	**5**	**6**	**7**		
Delta (δ)	73	17	99	93	46	8	70		
Distance (s,t)	**Return values or no information**							**Delta distance**	**Actual distance**
(1,2)	18	17	∞	∞	∞	∞	∞	18	1
(1,3)	103	102	99	101	100	∞	∞	103	4
(1,4)	95	94	∞	93	∞	∞	∞	95	2
(1,5)	49	48	∞	47	46	∞	∞	49	3
(1,6)	12	11	9	10	9	8	∞	12	4
(1,7)	75	74	∞	73	72	71	70	75	5
(2,3)	∞	102	99	101	100	∞	∞	102	3
(2,4)	∞	94	∞	93	∞	∞	∞	94	1
(2,5)	∞	48	∞	47	46	∞	∞	48	2
(2,6)	∞	11	9	10	9	8	∞	11	3
(2,7)	∞	74	72	73	72	71	70	74	4
(3,4)	∞	∞	95	93	94	95	∞	95	2
(3,5)	∞	∞	47	∞	46	47	∞	47	1
(3,6)	∞	∞	9	∞	9	8	∞	9	1
(3,7)	∞	∞	72	∞	72	71	70	72	2
(4,5)	∞	∞	∞	47	46	∞	∞	47	1
(4,6)	∞	∞	9	10	9	8	∞	10	2
(4,7)	∞	∞	72	73	72	71	70	73	3
(5,6)	∞	∞	9	∞	9	8	∞	9	1
(5,7)	∞	∞	72	∞	72	71	70	72	2
(6,7)	∞	∞	∞	∞	∞	71	70	71	1

In fact the return values approximate the delta distance quite well. However, this data does not provide insight into actual distances. In order to demonstrate the distortion of information we plotted delta distances against actual distances in Figure 7. The plot presents itself as very scattered. The correlation coefficient between both actual and delta distances is 0.14. The low correlation coefficient indicates there is barely any relationship in the data (0 would indicate no relationship). While clearly with larger graphs more data is available and more sophisticated information retrieval methods might be applicable, we yet validated that the decentral computation does not reveal relevant information – given our exemplary network. As stated before, we leave a systematic validation of larger graphs (real-world SCNs) for future research.

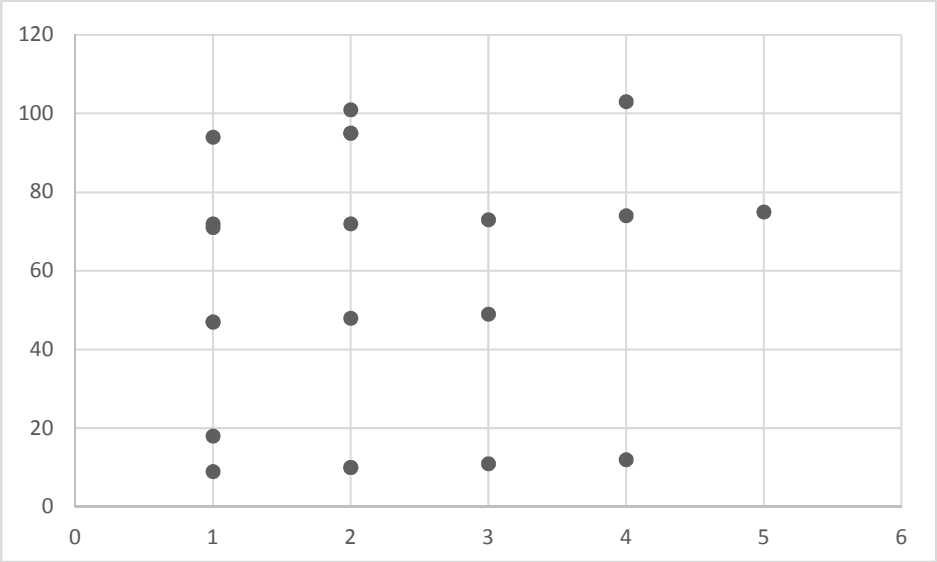

Figure 7 Delta distance versus actual distance

Additionally, players store information about shortest paths by design. Doing so they might infer additional information: in our exemplary network, player 5 knows that player 3 and 6 are neighbors because player 6 and 3 are 5's neighboring players and their shortest paths are not via player 5. Player 5 clearly knows the neighbor relationship right from initialization and stores shortest paths information.

Player 5 knows also that players 1 and 6, 2 and 6, as well as 4 and 6 are not neighboring players. The latter information is inferred based on the information that their shortest path is via player 5. But the player is not knowing their exact positioning and if there exists any other alternative shortest path.

It is to conclude that the gained information about the network's structure, even in a small network is limited. By increasing the network's size and complexity the possibility of inferring information decreases. Additionally, the inferred information on non-neighboring vertices is limited. This is similar to a common situation of a SCN. In reality, in a SCN, a firm knows more information about its neighbors. The firm can partially reveal information about the

neighbors of its own neighbors. By going further in the SCN, the firm is less capable to deduce the underlying topology or identity of the firms. Moreover, in most of the SCNs, there are some main players that are known by everyone. If other firms identify these firms and their importance, it is not a risk for these players. Their importance and positioning in the network is predictable for most of the firms in the SCN.

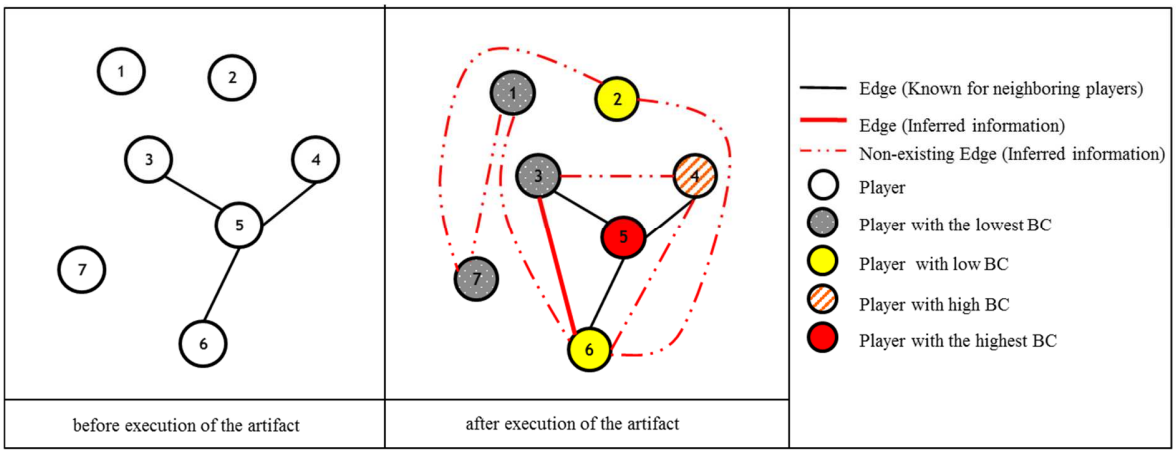

Figure 8 The network's structure from player 5's perspective [27]

5.1 Discussion

While this paper's evaluation represents a first step toward subsequent real-world evaluation steps, we discuss security challenges which might arise on the road ahead from a multiple layer perspective as a basis for future research. The layers comprise the orchestration platform i.e. the ISA, the algorithms, and players' behavior.

First, it is important to reemphasize that risk management as stated before will greatly benefit, if players in a SCN can determine the systemic risk they are exposed to. However, the necessary disclosure of relevant information is subject to the player's trust in the network. Players will not share information, if they fear that the underlying platform, algorithms and other players' behavior are unfit to maintain high standards of security.

Regarding the ISA, the paper suggests either an ERP module or a blockchain-based instance. Both have advantages and disadvantages alike with regard to security. The latter might facilitate a decentral execution based on cryptographic protocols but might require additional information to be exchanged which ought to be analyzed when there is a first prototypical implementation – e.g. as part of a future research contribution. The former corresponds to a central execution platform and allows for an implementation and information exchange exactly as presented herein. However, technically it might be difficult to shield the orchestration from a curious ISA operator (which is considered to be a player for itself). Other players might be skeptical and decline requests to participate in the network and its corresponding information disclosure.

Regarding the security of the algorithms, we performed a critical evaluation and pointed out that we suggest to perform further evaluation steps on large-scale networks. The SCM algorithms we apply have been proven to satisfy security

standards. The protocols either rely on some mathematical problem such as factoring or are unconditional referring to a probability of error which can be sized arbitrarily small (e.g. [25, 26]).

Finally, security in the SCN is challenged by the behavior of the players itself. As this papers assumes semi-honest players which do not deviate from the protocol - i.e. the set of methods and its orchestration - real world players might prove to be malicious for e.g. business strategic reasons. From a social science perspective, it might be relevant to better understand motifs of players to do so but from a computer networks perspective, it should be even more tempting to advance the methods of this contribution to prove security under malicious environments featuring players which are likely to cheat. While those preserve privacy in any case, it will be interesting to observe, if honest players would then abort the execution resulting in little information on systemic risk of individual players. Moreover, increasing security, comes at the cost of efficiency as already pointed out in our evaluation. MPC algorithms for semi-honest environments might be considered relatively efficient. Corresponding algorithms for malicious environments might be secure but too inefficient for use in practice. Developing resource-efficient algorithms to deal with malicious environments poses a great but demanding opportunity for the field of security in computer networks.

Summarizing, we state that security in general and privacy preserving computing in specific enable improvements of risk management in SCNs but the interaction of the various layers as described above leave unsolved challenges for research, this contribution cannot solve at once.

6. Conclusion

In this paper, we proposed an artifact which preserves privacy and identifies the risky players in the SCNs applying the BC measure. Based on the guidelines of [31], and [32] for conducting design science research, we can summarize our work as follows: Our artifact consists of four main methods. It is an exaptation solution, because we adopted the existing methods in social networks and cryptography algorithms to identify risks in SCNs. Our artifact is formally noted and therefore is well-defined. Based on the literature (e.g. [35]) we addressed two relevant problems: the risk identification in SCNs and privacy concerns of firms in SCNs. We focused on the study of [15] and decided to calculate the BC as a measure to identify risky firms. In the evaluation section, beside the testing and descriptive evaluation, we illustrated that in our artifact, even in a small exemplary network, the inferred information is limited. To develop a rigorous artifact, we applied well established methods of other fields and extended them to our problem context. Regarding the contribution of our result, we choose the evolving technical solutions in computer science and network theory, to answer the question of risk management in SCNs.

In this paper, we focused on identifying risks and kept the information as private as possible. However, higher visibility in the network facilitates improved risk management [67]. Therefore, it might be necessary that firms agree on sharing more information than the BCs. For instance they might decide to reveal the identities of firms with the BC among top 10%, because they are the most risky ones for the network. On the one hand the more information is shared, the highest

is the privacy at risk, and on the other hand it is inevitable to share extra information to reach the network's robustness. Hence, the firms in the network should deal with the trade-off between sharing additional information to facilitate risk management in the network or preserve their privacy.

Although the BC measure identifies the risks in the SCN, integration of complementary network analysis approaches (e.g. [18]) in our artifact for an enhanced risk identification, is subject to further research. It is also important to study the intensity of connection and their impacts on the network. These subjects as well as improvement of computational complexity are subject to further research.

7. Appendix

Table 8 Description of the commonly used variables

Variable	Mathematical Variable	Description
$sourcePlayerNumber$: int	s	Denotes the unique number of the source player.
$targetPlayerNumber$: int	t	Denotes the unique number of the target player.
$currentPlayerNumber$: int	p	Denotes the unique number of the current instance of the class Player.
$neighboringPlayerNumber$: int	a	Denotes the unique number of a neighboring player.
$aPlayerNumber$: int	v	Denotes the unique number of a given player.

Table 9 Description of the variable defined for calculateSecureShortestPath() method

Variable	Mathematical Variable	Description
$temporary\ Distance$: int	td	Denotes a temporary variable saving the distances during calculation of the shortest paths. This variable ensure data consistency.

8. DATA AVAILABILITY

All data arising from this study are contained within the manuscript.

9. CONFLICTS OF INTEREST

The author declare that there is no conflict of interest regarding the publication of this article.

10. FUNDING STATEMENT

This research was (in part) carried out in the context of the Project Group Business and Information Systems Engineering of the Fraunhofer Institute for Applied Information Technology FIT.

Grateful acknowledgement is due to the DFG (German Research Foundation) for their grant [ITPM (FR 2987/2-1, BU 809/13-1)] making this paper possible.

11. REFERENCES

[1] H. Peck, *Creating Resilient Supply Chains: A Practical Guide*, Cranfield University, United Kingdom, 2003.

[2] S. M. Wagner and N. Neshat, "A comparison of supply chain vulnerability indices for different categories of firms," *International Journal of Production Research*, vol. 50, no. 11, pp. 2877–2891, 2012.

[3] D. Helbing, "Globally networked risks and how to respond," *Nature*, vol. 497, no. 7447, pp. 51–59, 2013.

[4] D. Acemoglu, A. Ozdaglar, and A. Tahbaz-Salehi, *Networks, Shocks, and Systemic Risk*, National Bureau of Economic Research, Cambridge, MA, 2015.

[5] C. Ellinas, N. Allan, and A. Johansson, "Project systemic risk: Application examples of a network model," *International Journal of Production Economics*, vol. 182, pp. 50–62, 2016.

[6] HBR Advisory Council, "Is Your Supply Chain Sustainable?," *Harvard Business Review;*, vol. 88, no. 10, p. 74, 2010.

[7] World Economic Forum, "Global Risks 2008: A Global Risk Network Report," 2008,

[8] G. Fridgen, C. Stepanek, and T. Wolf, "Investigation of exogenous shocks in complex supply networks–a modular Petri Net approach," *International Journal of Production Research,* ahead-of-print, pp. 1–22, 2014.

[9] M. Giannakis and M. Louis, "A multi-agent based framework for supply chain risk management," *Journal of Purchasing and Supply Management*, vol. 17, no. 1, pp. 23–31, 2011.

[10] L. K. Chu, Y. Shi, S. Lin et al., "Fuzzy chance-constrained programming model for a multi-echelon reverse logistics network for household appliances," *Journal of the Operational Research Society*, vol. 61, no. 4, pp. 551–560, 2010.

[11] C. Blome and T. Schoenherr, "Supply chain risk management in financial crises—A multiple case-study approach," *International Journal of Production Economics*, vol. 134, no. 1, pp. 43–57, 2011.

[12] T. Y. Choi and Y. Hong, "Unveiling the structure of supply networks: case studies in Honda, Acura, and DaimlerChrysler," *Journal of Operations Management*, vol. 20, no. 5, pp. 469–493, 2002.

[13] R. Wilding, J. Miemczyk, T. E. Johnsen et al., "Sustainable purchasing and supply management: A structured literature review of definitions and measures at the dyad, chain

and network levels," *Supply Chain Management: An International Journal*, vol. 17, no. 5, pp. 478–496, 2012.
[14] K. Zhao, A. Kumar, T. P. Harrison et al., "Analyzing the resilience of complex supply network topologies against random and targeted disruptions," *Systems Journal, IEEE*, vol. 5, no. 1, pp. 28–39, 2011.
[15] Y. Kim, T. Y. Choi, T. Yan et al., "Structural investigation of supply networks: A social network analysis approach," *Journal of Operations Management*, vol. 29, no. 3, pp. 194–211, 2011.
[16] A. Vereecke, R. van Dierdonck, and A. de Meyer, "A typology of plants in global manufacturing networks," *Management science*, vol. 52, no. 11, pp. 1737–1750, 2006.
[17] K. J. Mizgier, M. P. Jüttner, and S. M. Wagner, "Bottleneck identification in supply chain networks," *International Journal of Production Research*, vol. 51, no. 5, pp. 1477–1490, 2013.
[18] M. E. J. Newman, *Networks*: *An introduction*, Oxford Univ. Press, Oxford, 2013.
[19] L. C. Freeman, "A Set of Measures of Centrality Based on Betweenness," *Sociometry*, vol. 40, no. 1, p. 35, 1977.
[20] S. Wasserman and K. Faust, *Social network analysis*: *Methods and applications*, Cambridge Univ. Press, Cambridge, 2009.
[21] J. Hallikas, I. Karvonen, U. Pulkkinen et al., "Risk management processes in supplier networks," *International Journal of Production Economics*, vol. 90, no. 1, pp. 47–58, 2004.
[22] W. Kersten, P. Hohrath, and M. Winter, "Risikomanagement in Wertschöpfungsnetzwerken–Status quo und aktuelle Herausforderungen," *Supply Chain Risk Management*, p. 7, 2008.
[23] U. Brandes, "A faster algorithm for betweenness centrality*," *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.
[24] Y. V. Hochberg, A. Ljungqvist, and Y. Lu, "Whom you know matters: Venture capital networks and investment performance," *The Journal of Finance*, vol. 62, no. 1, pp. 251–301, 2007.
[25] A. C. Yao, "How to generate and exchange secrets," in *27th Annual Symposium on Foundations of Computer Science*, pp. 162–167, IEEE, 1986.
[26] O. Goldreich, S. Micali, and A. Wigderson, "How to Play any Mental Game - A Completeness Theorem for Protocols with Honest Majority," in *nineteenth annual ACM Symposium on the Theory of Computing,* A. V. Aho, Ed., pp. 218–229, 1987.
[27] G. Fridgen and T. Zare Garizy, "Supply Chain Network Risk Analysis: A Privacy Preserving Approach," in *23rd European Conference on Information Systems (ECIS 2015)*, 2015.
[28] Tirazheh Zare Garizy, "Systemic Risk Assessment in Complex Network Structures: Information Technology as a Challenge and a Chance,".
[29] T. Y. Choi and D. R. Krause, "The supply base and its complexity: implications for transaction costs, risks, responsiveness, and innovation," *Journal of Operations Management*, vol. 24, no. 5, pp. 637–652, 2006.
[30] D. R. Lessard, "Uncertainty and Risk in Global Supply Chains," *MIT Sloan Research Paper No. 4991-13*, 2013.
[31] A. R. Hevner, S. T. March, J. Park et al., "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[32] S. Gregor and A. R. Hevner, "POSITIONING AND PRESENTING DESIGN SCIENCE RESEARCH FOR MAXIMUM IMPACT," *MIS Quarterly*, vol. 37, no. 2, 337-A-6, 2013.
[33] J. Blackhurst, T. Wu, and P. O'grady, "Network-based Approach to Modelling Uncertainty in a Supply Chain," *International Journal of Production Research*, vol. 42, no. 8, pp. 1639–1658, 2004.
[34] M. Arns, M. Fischer, P. Kemper et al., "Supply Chain Modelling and Its Analytical Evaluation," *Journal of the Operational Research Society*, vol. 53, no. 8, pp. 885–894, 2002.
[35] H. U. Buhl and H.-G. Penzel, "The Chance and Risk of Global Interdependent Networks," *Business & Information Systems Engineering*, vol. 2, no. 6, pp. 333–336, 2010.
[36] V. Babich, A. N. Burnetas, and P. H. Ritchken, "Competition and diversification effects in supply chains with supplier default risk," *Manufacturing & Service Operations Management*, vol. 9, no. 2, pp. 123–146, 2007.
[37] J. F. Yates and E. R. Stone, "The risk construct," in *Risk-taking behavior,* J. F. Yates, Ed., pp. 49–85, John Wiley & Sons, New York, 1992.
[38] T. Gyorey, M. Jochim, and S. Norton, "The challenges ahead for supply chains," *McKinsey on Supply Chain: Select Publications*, pp. 10–15, 2011.
[39] M. A. Bellamy and R. C. Basole, "Network Analysis of Supply Chain Systems: A Systematic Review and Future Research," *Systems Engineering*, vol. 16, no. 2, pp. 235–249, 2013.
[40] M. E.I. Li and T. Y. Choi, "Triads in Services Outsourcing: Bridge, Bridge Decay and Bridge Transfer*," *Journal of Supply Chain Management*, vol. 45, no. 3, pp. 27–39, 2009.
[41] A. C. Yao, "Protocols for secure computations," in *23rd Annual Symposium on Foundations of Computer Science*, pp. 160–164, 1982.
[42] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[43] R. Sheikh, B. Kumar, and D. K. Mishra, "Privacy Preserving k Secure Sum Protocol," *International Journal of Computer Science and Information Security*, vol. 6, no. 2, pp. 184–188, 2009.
[44] R. Cramer, I. Damgard, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing: An Information Theoretic Approach*, Aarhus University, Aarhus Unoversity, Denmark, 2013.
[45] P. Bogetoft, I. Damgård, T. Jakobsen et al., "A Practical Implementation of Secure Auctions Based on Multiparty Integer Computation," *Financial Cryptography and Data Security*, vol. 4107, pp. 142–147, 2006.
[46] E. A. Abbe, A. E. Khandani, and A. W. Lo, "Privacy-preserving Methods for Sharing Financial Risk Exposures," *The American Economic Review*, vol. 102, no. 3, pp. 65–70, 2012.
[47] F. Kerschbaum, "Secure and sustainable benchmarking in clouds," *Business & Information Systems Engineering*, vol. 3, no. 3, pp. 135–143, 2011.
[48] F. Kerschbaum, A. Schroepfer, A. Zilli et al., "Secure Collaborative Supply-Chain Management," *Computer*, vol. 44, no. 9, pp. 38–43, 2011.
[49] D. Dolev and A. C. Yao, "On the security of public key protocols," *Information Theory, IEEE Transactions on*, vol. 29, no. 2, pp. 198–208, 1983.

[50] D. Beaver, S. Micali, and P. Rogaway, “The Round Complexity of Secure Protocols,” in *twenty-second annual ACM symposium on Theory of computing,* H. Ortiz, Ed., pp. 503–513, ACM, 1990.
[51] Y. Lindell and B. Pinkas, “A proof of security of Yao’s protocol for two-party computation,” *Journal of Cryptology*, vol. 22, no. 2, pp. 161–188, 2009.
[52] T. I. Reistad, “Multi-party secure position determination,” in *Norsk Informatikkonferanse NIK 2006*, pp. 137–142, Norwegian University of Science and Technology, Trondheim, 2012.
[53] R. Jacob, D. Koschützki, K. A. Lehmann et al., “Algorithms for Centrality Indices,” in *Network Analysis,* D. Hutchison, T. Kanade, J. Kittler et al., Eds., vol. 3418, pp. 62–82, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
[54] D. J. Klein, “Centrality measure in graphs,” *Journal of mathematical chemistry*, vol. 47, no. 4, pp. 1209–1223, 2010.
[55] T. H. Cormen, C. E. Leiserson, R. L. Rivest et al., *Introduction to algorithms*, MIT press Cambridge, 2001.
[56] E. F. Moore, “The shortest path through a maze,” in *Proceedings of the International Symposium on the Theory of Switching,* Harvard University Press, Ed., pp. 285–292, Bell Telephone System, 1959.
[57] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.
[58] N. Edmonds, T. Hoefler, and A. Lumsdaine, “A space-efficient parallel algorithm for computing betweenness centrality in distributed memory,” in *International Conference on High Performance Computing (HiPC)*, pp. 1–10, IEEE, 2010.
[59] R. W. Floyd, “Algorithm 97: shortest path,” *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.
[60] S. Warshall, “A theorem on boolean matrices,” *Journal of the ACM (JACM)*, vol. 9, no. 1, pp. 11–12, 1962.
[61] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2009.
[62] J. Brickell and V. Shmatikov, “Privacy-preserving graph algorithms in the semi-honest model,” *Advances in Cryptology-ASIACRYPT 2005*, pp. 236–252, 2005.
[63] R. Canetti, “Theory of cryptography,” in *Fifth theory of cryptography conference, TCC,* R. Canetti, Ed., Springer, 2008.
[64] Y. Huang, J. Katz, and D. Evans, “Quid-pro-quo-tocols: Strengthening semi-honest protocols with dual execution,” in *Security and Privacy (SP), 2012 IEEE Symposium on*, pp. 272–284, IEEE, 2012.
[65] T. Schneider, *Engineering Secure Two-Party Computation Protocols*: *Design, Optimization, and Applications of Efficient Secure Function Evaluation*, Springer, Berlin, Heidelberg, 2012.
[66] T. G. Gill and A. R. Hevner, “A fitness-utility model for design science research,” *ACM Transactions on Management Information Systems (TMIS)*, vol. 4, no. 2, p. 5, 2013.
[67] R. C. Basole and M. A. Bellamy, “Supply Network Structure, Visibility, and Risk Diffusion: A Computational Approach,” *Decision Sciences*, vol. 45, no. 4, pp. 753–789, 2014.