# Privacy-Preserving Social Network Analysis for Criminal Investigations

Florian Kerschbaum
SAP Research
Karlsruhe, Germany
florian.kerschbaum@sap.com

Andreas Schaad
SAP Research
Karlsruhe, Germany
andreas.schaad@sap.com

## ABSTRACT

Social network analysis (SNA) is now a commonly used tool in criminal investigations, but evidence gathering and analysis is often restricted by data privacy laws. We consider the case where multiple investigators want to collaborate, but do not yet have sufficient evidence that justifies a plaintext data exchange. This paper proposes a solution for privacy-preserving social network analysis where several investigators can collaborate without actually exchanging sensitive private information. An investigator can request data from other sites to augment his view without revealing personally identifiable data. The investigator can compute important metrics by means of a SNA on the subject while keeping the entire social network unknown him.

## Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications*; D.4.6 [**Operating Systems**]: Security and Protection—*Cryptographic controls*

## General Terms

Algorithms, Security

## Keywords

Privacy, Social Network Analysis, Criminal Investigations, Data Sharing

## 1. INTRODUCTION

In federated states or organization of states, such as the European Union or the United States, a common approach to organized crime is necessary. For this purpose, federal law enforcement agencies, such as Europol or the FBI, have been established. Nevertheless, data privacy laws or simply data governance concerns restrict supplying institutions from sharing their data, unless there is a hard corroborating evidence on a case and subject under investigation. In particular, in the European Union [1], data privacy is regarded as a high social and political value and the dilemma on how to generate evidence without violating privacy laws is evident.

A common tool for the criminal investigator is social network analysis. It graphically depicts the suspects and their connections to other people or artifacts, such as telephone numbers or bank accounts, and allows the computation of certain metrics. Not all the facts composing the entire picture of a case may be known to one investigator. In particular, in pan-European organized crime, local police forces may only be aware of a partial view of the picture, as the case studies in the framework of the R4eGov project suggest [25].

This necessitates data exchange between the institutions, but European data privacy laws prohibits data exchange without reasonable cause and in excessive amounts. Therefore we propose a solution where the local investigator, or an investigator at the superordinate institution has access to all information, but without revealing sensitive or private details. This allows the investigator to still use SNA and profit from its achievements without breaking individual privacy rights or guidelines of other institutions.

Privacy-Preserving SNA has been suggested in the literature before, but we have found the solutions to be insufficient for the requirements of our scenario. In [15] a fully anonymized version of the social network is computed. This does not allow the investigator to track his suspect anymore and he cannot gain additional information or collect evidence about him. In [8] a special algorithm with privacy preservation for computing a metric within a network to be used as a recommendation value is proposed, but investigators are used to centrality metrics they are trained on, such as betweeness and closeness [26].

This paper contributes an algorithm to compute the important centrality metrics of betweeness and closeness without revealing personally identifying information and without revealing the entire social network.

The algorithm, called "Compute Metrics", provides higher privacy guarantees, as it does not even reveal the entire the social network (except its size), but still allows important metrics to be computed about the subject. These metrics allow the identification of the role the subject is playing within the criminal organization [26].

The remainder of the paper is structured as follows: The next Section reviews related work. This is followed by the algorithm: Computing metrics without revealing the network itself. The algorithm section is divided into building


*WPES'08,* October 27, 2008, Alexandria, Virginia, USA.

blocks, protocol description and analysis. This is followed by some remarks on practical implementations. The final section presents the conclusions of this work.

## 2. RELATED WORK

SNA has been used for criminal investigations for a long time [18, 23, 26]. Recent research [26] suggests using graphical tools and investigates the impact of SNA. We can conclude that SNA is a widely accepted tool in criminal investigations.

Privacy-Preserving SNA has been first proposed by [15]. They compute an anonymized graph of the social network, such that no one should be able to track their position in the graph. They allow for certain modifications of the correctness of the anonymized graph in order to prevent tracking of one's position. E.g. they may bound the number of incoming connections or apply similar restrictions.

While this provides strong privacy guarantees it does not match the requirements of our scenario. An investigator intends to gather *additional* information to his present view of the social network. It is therefore unacceptable to anonymize his view, but the goal is to augment it with additional information about the entire network.

Collaborative filtering is the process of collaboratively identifying "outstanding" information items. It is particularly useful for recommender systems in e-commerce where outstanding items corresponds to products likely to be bought. It is related to SNA, since both compute metrics for vertices in the network. Special collaborative filtering algorithms for graphs have been proposed in [8, 9, 22], but the research on SNA for criminal investigations [26] suggests that common metrics the investigators are used to, such as betweeness and closeness, are preferred.

Similarly a new algorithm for link analysis is proposed in [12]. Link analysis is a special collaborative filtering technique for a network or a graph of vertices. As a criticism, it applies here as well that practitioners prefer established metrics. It is important to note here, that different metrics require different protocols for computing them. In particular one can often optimize the performance of a privacy-preserving protocol by picking a metric that is easy to compute in a privacy-preserving manner, but that might be less useful to the user. Our focus is clearly on maximizing the added value to the investigator as an end-user of the system.

Privacy-Preserving SNA can be seen as a special case of secure multi-party computation (SMC) which can solve any distributed function privately. SMC has been suggested in [27] for the two-party case. The first multi-party solution have been suggested in [17] for the computational setting and in [4] for the information-theoretic setting. Efficient construction have been identified for different secret sharing schemes [6, 7]. Nevertheless, as in [15] stated, a straightforward application of these techniques would result in an unpractical protocol.

## 3. COMPUTING SNA METRICS

This protocol therefore computes the important SNA centrality metrics of betweeness and closeness without revealing the network. The betweeness centrality metric ranks vertices by the number of shortest paths that run through them and identifies vertices which connect strongly connected components. Vertices with a high betweeness metric indicate a gatekeeper functionality between two criminal organizations [26]. Let $\sigma_v(s,t) \in \{0,1\}$ be the number of shortest paths from $s$ to $t$ running through $v$. Betweeness is defined as

$$C_B(v) = \sum_{\substack{s,t \in V \\ s \neq t \; s \neq v \; t \neq v}} \frac{\sigma_v(s,t)}{(|V|-1)(|V|-2)}$$

The closeness centrality metric ranks vertices by their distance to all other nodes. Vertices with a high closeness metric indicate leadership in criminal organizations [26]. Let $\delta(v,t)$ be the length of the shortest path from $v$ to $t$ in $G$. Closeness is defined as

$$C_C(v) = \sum_{\substack{t \in V \\ v \neq t}} \frac{|V|-1}{\delta(v,t)}$$

The "Compute Metrics" algorithm computes betweeness and closeness without revealing the network structure.

### 3.1 Building Blocks

We use a commutative encryption scheme. In a commutative encryption scheme the order of encryption (with different keys) does not matter. We denote the encryption with Alice's key as $E_A()$ and with Bob's key as $E_B()$. Then, in a commutative encryption scheme, it holds that

$$E_A(E_B(x)) = E_B(E_A(x))$$

As we compare ciphertext, the encryption system cannot be semantically secure, but may be secret key. A candidate encryption system with all these properties is Pohlig-Hellman encryption [21]. Let $E^n()$ denote the encryption with all keys of the commutative encryption scheme, i.e. $E^n(x) = E_{X_1}(E_{X_2}(\ldots E_{X_n}(x)))$.

We also use a homomorphic, threshold encryption system. Let $E^*()$ denote encryption in this homomorphic, threshold encryption system. We require the homomorphic property to allow (modular) addition of the plaintexts. It then holds that

$$E^*(x)E^*(y) = E^*(x+y)$$

From which by simple arithmetic it follows that

$$E^*(x)^y = E^*(xy)$$

In a threshold encryption system the decryption key is replaced by a distributed protocol. Only if $t$ or more parties collaborate they can perform a decryption. No coalition of less than $t$ parties can decrypt a ciphertext. We require a collaboration of all parties, i.e. $t = n$ (since we operate in the semi-honest model and do not consider faults). Then a ciphertext can only be decrypted if all parties collaborate.

The homomorphic, threshold encryption system is furthermore public-key, i.e. any party can perform the encryption operation $E^*()$ (by itself).

The ciphertexts are semantically secure, i.e. their ciphertext reveals nothing about the plaintext. More precisely, the ciphertexts are indistinguishable under chosen plaintext attack (IND-CPA). This implies an important property of re-randomization where an input ciphertext is modified, such that it cannot be linked to its original anymore without modifying the plaintext. In our encryption system this is best

performed by "adding 0": $E^*(x)E^*(0) = \widehat{E}^*(x)$, but $E^*(x)$ and $\widehat{E}^*(x)$ are unlinkable without the decryption key.

An encryption system satisfying all our requirements has been described in [11], a variation of [20].

## 3.2 Protocol

In order to compute the two centrality metrics we need to compute the shortest path between all pairs of vertices. The shortest paths has been privately computed in [5], but they assume that the graph is public and only the weights are private. We need to keep the graph private as well.

We use the Floyd-Warshall algorithm [13] as basis for our computation. The Floyd-Warshall algorithm computes the all-pairs shortest path and is an example of dynamic programming. Privacy-preserving dynamic programming has been done in [3], but specifically for two parties while we consider the multi-party setting.

The Floyd-Warshall algorithm proceeds as follows

```
for k:= 1 to n
  for i:= 1 to n
    for j:= 1 to n
      M[i][j] = min(M[i][j], M[i][k] + M[k][j])
```

Initially the matrix $M$ contains the edges $E$, i.e. $M[i][j] = 1$ if $(v_i, v_j) \in E$ and $M[i][j] = \infty$ otherwise ($M[i][i] = 0$). Intuitively the algorithm checks whether a shorter path from $v_i$ to $v_j$ exists via $v_k$. At the end of the algorithm the matrix $M$ contains the length of the shortest path from $v_i$ to $v_j$ at its $i, j$-th position.

We keep the elements of the matrix $M$ encrypted under $E^*()$, such that no party individually has access to it. Every participant $X_i$ keeps a record of the current state of the matrix, i.e. the encryptions of each element. They then need to collaboratively engage in a "Minimum" protocol to compute the new element of the matrix. All participants engage in $|V|^3$ of these "Minimum" protocols following the Floyd-Warshall algorithm. One can use any multi-party "Minimum" protocol from the literature, e.g. [10, 24]. In the full version of the paper we propose a particularly fast protocol, but omit details here for brevity.

### 3.2.1 Computing the Initial Matrix

The parties can now compute the iteration step of the dynamic program, but the matrix $M$ must be initialized with the values from the set $E$ of edges.

First, the participants must agree on a common set $V$ of vertices. The vertex labels must not reveal identifiable information. Therefore, the participants must agree on an anonymized version of $V$. The anonymization for $v$ is computed using $E^n(v)$ by forwarding and encrypting the set of edges of each participant. The participants compute the set union of their anonymized local sets $V_i$. A set-union protocol does not reveal the overlaps in the input sets. Set-union protocols can be found in the literature in [5, 14, 19]. We propose a multi-party version of [5], since it uses a "Minimum" as used above. The overall protocol proceeds as follows:

1. First compute the pseudonyms without revealing the anonymized sets $V_i$. Note that the size of $V_i$ is revealed. This can be prevented by padding with random values.
   (a) Each participant $X_i$ encrypts his vertices $v_j \in V_i$ by $E_i()$ and $E'_i()$: $E_i(E'_i(v_j))$. He sends them to participant $X_{i+1}$.
   (b) Each participant $X_{i+1}$ doubly encrypts the received and already encrypted vertices with $E_{i+1}$: $E_{i+1}(E_i(E'_i(v_j)))$. He sends the result to participant $X_{i+2}$.
   (c) All participants $X_i$ repeat step 1b $n-2$ more times, such that each participant $X-i$ receives his initial values as $E^n(E'_i(v_j))$.
   (d) Each participant decrypts the received vertices with $D'_i()$ resulting in $E^n(v_j)$. Note that due to the commutative encryption the order of encryption does not matter.
2. Second compute the set-union of the anonymized vertices.
   (a) Each participant $X_i$ sorts his list of anonymized vertices $E^n(v_j)$ in ascending order of the ciphertext.
   (b) Each participant $X_i$ encrypts his minimum element $E^n(v_1)$ with $E^*()$ and sends $E^*(E^n(v_1))$ to $X_1$. He indicates the special element $\perp$, if he has no more pseudonyms.
   (c) All participants $X_i$ engage in the "Minimum" protocol for the inputs as described above.
   (d) All participants $X_i$ commonly decrypt the result $E^n(v_\gamma)$. Each participant that has the pseudonym $E^n(v_\gamma)$ in his list, removes it from this list.
   (e) They repeat steps 2b till 2d until all participants' lists are empty. Participants whose lists are empty use a top element $E^*(\top)$ outside of the domain of anonymized vertices as input. If this element $\top$ is computed as the minimum element the protocol ends, as now all lists are empty.

Each participant now holds the set $V$ of anonymized vertices $E^n(v)$ and knows the pseudonyms for his vertices $v$. He can then compute the anonymized set $E_i$ of his edges $(E^n(v), E^n(v'))$. The participants must now compute the matrix $M$. The rows and columns of $M$ each correspond to an anonymized vertex $E^n(v)$. Each participants sorts the set $V$ in lexicographically ascending order of the ciphertexts. All elements of $M$ can then be initialized with $\infty$ ($M[i][i] = 0$). Now each participant must set the values of $M$ corresponding to his edges $(E^n(v), E^n(v'))$ to 1. To achieve this they can run this protocol:

1. Participant $X_1$ prepares the initial matrix M with
$$M[i][j] = E^*(\infty) \forall i, j \in \{1, \ldots, |V|\}$$
$$M[i][i] = E^*(0) \forall i \in \{1, \ldots, |V|\}$$
2. Participant $X_k$ ($k = 1$) replaces the entries for edges in $M$ with $E^*(1)$.
$$M[i][j] = E^*(1) \forall (i, j) \in E_k$$
He rerandomizes all ciphertexts and sends the result to $X_{k+1}$.
3. Each participant $X_k$ ($k = 2, \ldots, n$) performs step 2.

4. Each participant $X_k$ $(k = 1, \ldots, n-2)$ keeps a record of the matrix $M$ and forwards it to $X_{k+1}$.

The algorithm is correct, since all edges in $E$ will be set and duplicates are not noticed, since no participant knows whether he replaces a 1 or a $\infty$ with his 1.

The participants have then successfully initialized the matrix $M$ and can run the dynamic program of the Floyd-Warshall algorithm. They compute the minimum for each loop iteration by choosing the corresponding matrix element and homomorphically computing the sum of the other two elements. In the end, they will end up with an encrypted version of the matrix $M$ that has plaintexts of the length of the shortest paths between all pairs of vertices.

#### 3.2.2 Computing the Metrics

So far we have computed the shortest-path matrix in the Floyd-Warshall algorithm, but our goal is to compute the centrality metrics of closeness and betweeness for one vertex, a suspect. Let $X_1$ be the investigator who has a suspect he wants query in the matrix. The other participants $X_2, \ldots, X_n$ should not learn which is the queried vertex. They can do so using the following protocols.

##### 3.2.2.1 Closeness.

Let $v_s$ be the vertex for which intelligence is gathered. The investigator $X_1$ selects the row $s$ for vertex $v_s$ in $M$ and computes

$$c_c = \prod_{i=1}^{m} M[s][i] = E^*(\sum_{i=1}^{m} \delta_{s,i})$$

$X_1$ rerandomizes $c_c$ to $c'_c$ and distributes it to all participants. All participants jointly decrypt $c'_c$ and $X_1$ can compute

$$C_C(v_s) = \frac{|V| - 1}{D^*(c'_c)}$$

##### 3.2.2.2 Betweeness.

Betweeness is more complicated to compute. $X_1$ needs to calculate the number of shortest paths through $v_s$. He keeps a second matrix $T$ of size $|V| \times |V|$. All entries of $T$ are encrypted with $E^*()$, just as those of $M$. He initializes the matrix $T$ at the beginning of the "Compute SNA Metrics" protocol as follows

$$\begin{aligned} T[s][j] &= E^*(1) && \forall j \neq s \\ T[i][j] &= E^*(0) && \text{otherwise} \end{aligned}$$

In each "Minimum" protocol for $M[i][j]$ in the Floyd-Warshall algorithm he augments the messages with the corresponding elements of $T$, i.e. in the message field for $M[i][j]$ he adds another field $T[i][j]$ and in the message field for $M[i][k]+M[k[j]$ he add a field $T[i][k] + T[k][j]$. He computes the addition of the plaintexts by the multiplication of the ciphertexts due their homomorphic property and the ciphertexts are re-randomized by all other parties, such that $X_1$ cannot track the entries. After each "Minimum" protocol he updates $T[i][j]$ with the value from the field in the minimum index $\gamma$.

At the end of the Floyd-Warshall algorithm $T[i][j] \in 0, 1$ indicates with a 1 if the path from $v_i$ to $v_j$ is via $v_s$. From the computation of $T[i][j]$ during the iteration it follows that it is an invariant of the algorithm that $T[i][j]$ equals the number of times the path from $v_i$ to $v_j$ crosses $v_s$ as an intermediary (i.e. all except the destination) vertex. Since the Floyd-Warshall algorithm computes the shortest path at its completion, no path can cross $v_s$ more than one time in the final matrix $T$.

After the completion of the Floyd-Warshall algorithm $X_1$ computes

$$\begin{aligned} c_b &= \prod_{i=1,\ldots,s-1,s+1,\ldots,m} \prod_{i=1,\ldots,s-1,s+1,\ldots,m} T[i][j] \\ &= E^*( \sum_{i=1,\ldots,s-1,s+1,\ldots,m} \sum_{i=1,\ldots,s-1,s+1,\ldots,m} \sigma_{i,j}) \end{aligned}$$

Participant $X_1$ rerandomizes $c_b$ to $c'_b$ and distributes it to all participants. All participants jointly decrypt $c'_b$ and $X_1$ can compute

$$C_B(v_s) = \frac{D^*(c'_b)}{(|V| - 1)(|V| - 2)}$$

### 3.3 Analysis

The "Compute SNA Metrics" protocol operates in the semi-honest setting. We strongly argue that this is appropriate for our application, since we are concerned with cooperating police organizations and officers whose main concern is protecting the privacy of the suspects and keeping practical data governance. That is, the organizations are inclined to follow the protocol, since their objective is not only the outcome of the collaborations, but also the process of data privacy protection. Since interest in collaboration can be assumed, the organizations could simply exchange data by bypassing the protocol, if they were not interested in data protection.

The proof of security is standard and follows the methodology of [16] by giving a simulator for the views of the participants. No information about the graph $G$ except the centrality metrics is leaked. This follows from the correct implementation of the functionality, i.e. the function implemented by the "Compute SNA Metrics" protocol is to just compute the two metrics, closeness and betweeness, for one vertex in question. The "Compute SNA Metrics" protocol therefore offers a high degree of privacy, but it limits the result of the computation.

## 4. IMPLEMENTATION ISSUES

### 4.1 Synonyms

Encryption is sensitive to spelling and capitalization mistakes. Therefore a person with two almost identical synonyms results in two very different ciphertexts and therefore vertices. This is a known problem and can be solved existing software solutions [2]. Such solutions provide unique identifiers by expanding names into all possible synonyms. Note that, there exists privacy-preserving extensions to these technologies using cryptographic hashing, but they do not extend to SNA as required in our case.

## 5. CONCLUSION

Social Network Analysis is becoming an important tool for investigators, but all the necessary information is often

distributed over a number of sites. Privacy legislation and data governance concerns prohibit freely sharing the information.

We have presented a protocol that allow the selective disclosure of information for Social Network Analysis. It only discloses the results of Social Network Analysis: two important centrality metrics. It thereby allows an investigator to gather intelligence on a suspect by querying remote data sources without disclosing even anonymized data.

This shows that Social Network Analysis can be used in a privacy-preserving manner by investigators. We present this information to European authorities in the context of the R4eGov project and hope it will serve both, the security of the people and the privacy necessary for freedom in the European Union.

## 6. REFERENCES


[1] Directive 95-46-EC on the protection of individuals with regard to the processing of personal data and on the free movement of such data. Available at *http://ec.europa.eu/justice_home/fsj/privacy*, 1995.

[2] IBM Entity Analytic Solutions. Available at *http://ibm.com/db2/eas*, 2005.

[3] M. Atallah, F. Kerschbaum, and W. Du. Secure and Private Sequence Comparisons. *Proceedings of the 2nd annual Workshop on Privacy in the Electronic Society*, 2003.

[4] M. Ben-Or, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. *Proceedings of the 20th annual ACM symposium on Theory of computing*, 1988.

[5] J. Brickell, and V. Shmatikov. Privacy-Preserving Graph Algorithms in the Semi-honest Model. *Proceedings of AsiaCrypt*, 2005.

[6] R. Cramer, I. Damgard, and U. Maurer. General Secure Multi-party Computation from any Linear Secret-Sharing Scheme. *Proceedings of EuroCrypt*, 2000.

[7] R. Cramer, I. Damgard, and J. Nielsen. Multiparty Computation from Threshold Homomorphic Encryption. *Proceedings of EuroCrypt*, 2001.

[8] J. Canny. Collaborative Filtering with Privacy. *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.

[9] J. Canny. Collaborative Filtering with Privacy via Factor Analysis. *Proceedings of the 25th International ACM Conference on Research and Development in Information Retrieval*, 2002.

[10] I. Damgard, M. Fitzi, E. Kiltz, J. Nielsen, and T. Toft. Unconditionally Secure Constant-Rounds Multi-party Computation for Equality, Comparison, Bits and Exponentiation. *Proceedings of Theoretical Cryptography Conference*, 2006.

[11] I. Damgard, and M. Jurik. A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. *Proceedings of Public Key Cryptography*, 2001.

[12] Y. Duan, J. Wang, M. Kam, and J. Canny. Privacy Preserving Link Analysis on Dynamic Weighted Graph. *Computational & Mathematical Organization Theory 11(2)*, 2005.

[13] R. Floyd. Algorithm 97: Shortest Path. *Communications of the ACM 5(6)*, 1962.

[14] K. Frikken. Privacy-Preserving Set Union. *Proceedings of Applied Cryptography and Network Security*, 2007.

[15] K. Frikken, and P. Golle. Private Social Network Analysis: How to Assemble Pieces of a Graph Privately. *Proceedings of the ACM Workshop on Privacy in the Electronic Society*, 2006.

[16] O. Goldreich. Secure Multi-party Computation. Available at *www.wisdom.weizmann.ac.il/~oded/pp.html*, 2002.

[17] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. *Proceedings of the 19th annual ACM conference on Theory of computing*, 1987.

[18] W. Harper, and D. Harris. The application of link analysis to police intelligence. *Human Factors 17(2)*, 1975.

[19] L. Kissner, and D. Song. Privacy-Preserving Set Operations. *Proceedings of CRYPTO*, 2005.

[20] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. *Proceedings of EUROCRYPT*, 1999.

[21] S. Pohlig, and M. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory 24*, 1978.

[22] H. Polat, and W. Du. Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques. *Proceedings of the 3rd IEEE International Conference on Data Mining*, 2003.

[23] M. Sparrow. The application of network analysis to criminal intelligence: an assessment of the prospects. *Social Networks 13*, 1991.

[24] T. Toft. Primitives and Applications for Multi-party Computation. *PhD dissertation, University of Aarhus*, 2007.

[25] T. Van Cangh, A. Boujraf. The Eurojust-Europol Case Study. Available at http://www.r4egov.eu/resources/details.php?Id_taxonomy=6, 2007.

[26] J. Xu, and H. Chen. Criminal Network Analysis and Visualization. *Communications of the ACM 48(6)*, 2005.

[27] A. Yao. Protocols for Secure Computations. *Proceedings of the annual IEEE Symposium on Foundations of Computer Science* 23, 1982.