Privacy-Preserving Graph Algorithms in the Semi-honest Model

Software Design Document

By Aviad Gilboa,Yaakov Khodorkovski , Dolev Dublon ,and Daniel Zaken
Date: (08/01/2023)

1. INTRODUCTION

1.1 Purpose

This Software Design Document (SDD) has been prepared to delineate the project's comprehensive architecture and system design titled "Privacy-Preserving Graph Algorithms in the Semi-honest Model". It offers a detailed view of the system's functionalities, design, and architecture. The primary target audience of this document is researchers and academics focusing on the field of multi-party computation in cryptography due to the fact this field has a lot of papers done but not a lot of implementations available.

One of the key goals of this project is to provide solutions that cater to scenarios that demand the implementation of privacy-preserving algorithms. For instance, telecommunication companies considering a merger or alliance might require a means to compute various algorithms on a combined network graph, without exposing sensitive information about their individual network infrastructures. Similarly, shipping companies could apply these algorithms to evaluate shipping capacities between different cities, utilizing the combined graphs of two or more companies, while preserving the confidentiality of individual shipping routes.

1.2 Scope

This SDD provides an overview of the proposed software system designed to implement privacy-preserving graph algorithms as delineated in the referenced paper. The objectives of this project are to provide a secure, efficient, and privacy-preserving system for performing APSD, SSSD, and privacy-preserving set union on two input graphs. The benefits include secure multi-party computation, preservation of privacy, and application in various data-sharing scenarios without revealing sensitive information.

1.3 Overview

This document presents a comprehensive description of our software project, "Privacy-Preserving Graph Algorithms in the Semi-honest Model". It's laid out in a clear and concise manner to provide an understanding of the system's architecture, data design, and user interface. It aims to provide readers with an easy-to-understand guide to the project's structure and design.

1.4 Reference Material

The core reference material for this document is the paper "Privacy-Preserving Graph Algorithms in the Semi-honest Model" by Justin Brickell and Vitaly Shmatikov.

1.5 Definitions and Acronyms

This section defines terms, acronyms, and abbreviations critical for comprehending the SDD. These include APSD (All Pairs Shortest Distance), SSSD (Single Source Shortest Distance), and more. The interpretation of these terms is crucial to understanding the design and functionality of the system under discussion.

2. SYSTEM OVERVIEW

Our project, "Privacy-Preserving Graph Algorithms in the Semi-honest Model", is designed to securely compute graph algorithms between two parties - Alice and Bob, in this case - while preserving the privacy of each party's data. Alice and Bob are not physical entities but rather symbolic representations of the two parties involved in the communication.

The software follows a client-server model, with Alice functioning as the server and Bob as the client. It's important to note that these designations are arbitrary and used purely for the purpose of defining roles within the network structure. They don't suggest any hierarchical relationship between the parties.

The system is based on the protocols described in the referenced academic paper, and it incorporates security measures to ensure the privacy of each party's data during computation. These protocols are designed to enable the computation of graph algorithms without revealing any more information about the parties' respective graphs than what is disclosed by the algorithm's output.

The software is designed to run on local machines between two computers and not on cloud-based platforms. It employs Python's TCP server for network communication. A clean handshake mechanism is put in place to confirm the successful exchange of data and ensure no data loss during transmission.

One of the key objectives of this project is to demonstrate how the privacy-preserving algorithms presented in the academic paper can be implemented in real-world software applications. The software is open-source and freely available, providing developers with a valuable resource that can be incorporated into their projects to perform privacy-preserving computations on graph data.

3. SYSTEM ARCHITECTURE

The system architecture is structured to achieve privacy-preserving computation of graph algorithms between two parties, adhering to the semi-honest model. Here, we provide a broad overview of how the responsibilities of the system are divided and then delegated to different components.

3.1 Architectural Design

The architectural design of the system is modular and relationship-oriented. It includes two main components: the Server Module (Alice) and the Client Module (Bob). These modules are designed to interact with each other over a secure network channel, with each module fulfilling its own unique set of responsibilities.

Alice, serving as the server, is responsible for setting up and maintaining the TCP server, initiating the secure handshake, and following the algorithmic protocol outlined in the "Privacy-Preserving Graph Algorithms in the Semi-honest Model" paper. Alice's responsibilities also include ensuring that the communication is kept secure and data integrity is maintained.

Bob, acting as the client, connects to Alice's server and participates in the handshake process. Once the connection is established and verified, Bob also adheres to the algorithmic protocol, ensuring privacy-preserving computation of graph algorithms.

Both Alice and Bob contribute to achieving the privacy-preserving computation by maintaining the secrecy of their input data and only disclosing the information required by the output of the algorithm.

3.2 Decomposition Description

The functionality of the system is subdivided into several key components: Network setup and connection, Secure handshake, Protocol implementation, Secure Bitwise OR computation, Tree Pruning method, and Result computation.

Network setup and connection encompasses the creation of the TCP server by Alice and Bob's initiation of a connection to this server. The Secure handshake process authenticates this connection and ensures that data integrity is maintained throughout the interaction.

Protocol implementation involves the adherence to the steps outlined in the "Privacy-Preserving Graph Algorithms in the Semi-honest Model" paper, ensuring that computations occur in a privacy-preserving manner.

The Tree Pruning method is also implemented, which is used to optimize the computation of the set union and reduce the number of edges being processed, enhancing the system's efficiency.
A pivotal part of the system is the Secure Bitwise OR computation using an ElGamal cryptographic approach. This computation allows the secure calculation of the union function on the sets of edges of the two input graphs while preserving privacy.

Finally, the Result computation component outputs the results as dictated by the implemented graph algorithms. This component ensures that the outputs are valid and adhere to the privacy-preserving condition stipulated in the semi-honest model.

3.3 Design Rationale

The rationale behind this architecture is to ensure the efficient and secure computation of the graph algorithms in a privacy-preserving manner. The system aims to guarantee that the software is easy to use and modify, thereby promoting its use in various fields that require privacy-preserving computations. The modular design also allows for easy maintenance and possible future enhancements. Moreover, the choice of Python's TCP server ensures a reliable and well-tested foundation for network communication.

4. DATA DESIGN

4.1 Data Description

In the secure multi-party computation system, the primary data involved include:

- The input graphs: These are provided by two parties, Alice and Bob, each representing a network of nodes and edges.
- The binary representations of the sets: Each party's input graph is transformed into a binary representation for computation in the secure set union algorithm.
- Prime numbers: Large prime numbers, exceeding 30 bits, are randomly generated and validated using the Miller-Rabin primality test. They form the basis for the generation of public keys used in the ElGamal cryptographic approach.
- Public keys: These are created from the cyclic group of the prime number and used in the encryption and secure computation of the bitwise OR operation.
- Result of the computation: The output of the secure set union operation is given to both parties.

4.2 Data Dictionary

- Input graphs: Graphical data structures comprised of nodes (vertices) and edges (links), representing network data.
- Binary representation: The conversion of the sets (derived from input graphs) into binary form to facilitate the secure set union operation.
- Prime numbers: Positive integers greater than 1 that have no divisors other than 1 and themselves. They are generated randomly and exceed 30 bits for the secure computation.
- Public keys: Keys derived from a cyclic group of the prime number, used to encrypt data and perform secure computation in a privacy-preserving manner.
- Computation result: The final output after the secure set union operation, representing the combined set in a privacy-preserving way, delivered to both parties.

5. COMPONENT DESIGN

5.1 Protocol Handler Component

This module handles the implementation of the privacy-preserving protocol as presented in the paper. It guides Alice (the server) and Bob (the client) in following the secure multi-party computation paradigm. This includes the computation of all pairs shortest distance (APSD), single source shortest distance (SSSD), and privacy-preserving set union.

5.2 Graph Conversion Component

This component is responsible for transforming the input graphs provided by Alice and Bob into a binary representation. This binary representation of nodes and edges is necessary for secure computation in the system.

5.3 Encryption Component

The encryption module creates public keys based on prime numbers generated using the Miller-Rabin primality test. These keys are used to encrypt the input graphs of both parties in a secure and privacy-preserving manner. This component ensures the system operates under the semi-honest model, preserving the privacy of each party's input graph.

5.4 Bitwise Operation Component

This module is in charge of securely computing the bitwise OR operation using ElGamal cryptographic approach. By doing this, it contributes to the privacy-preserving set union operation.

5.5 Secure Set Union Component

This component takes the binary representations of the two input graphs, securely computes the set union operation, and prunes unnecessary branches using the tree pruning method. This process is executed in a privacy-preserving manner, ensuring that no additional information about each party's input graph is revealed.

5.6 Output Generation Component

Once all the computations are complete, this component ensures that the result of the computation – a privacy-preserved graph depicting the union of both input graphs – is generated and provided to both parties.

6. HUMAN INTERFACE DESIGN

6.1 Code Design and Documentation:

Our software is designed to be accessible and easy to use for developers. We have put great emphasis on clean, modular code, accompanied by comprehensive documentation. Comments are strategically placed throughout the code to provide a clear understanding of the flow and functionality.

6.2 API Interface and Troubleshooting:

An intuitive API interface streamlines interactions and facilitates the implementation process. If users encounter problems, detailed error messages have been incorporated into the software to aid in troubleshooting and ensure effective resolution.

6.3 User Guide and Community Support:

To further assist developers in getting started with the software, a user guide will provide step-by-step instructions for software setup and usage. A dedicated platform for community support will also be available, enabling developers to interact, ask questions, share experiences, and learn from each other.

7. REQUIREMENTS MATRIX

The requirements matrix maps the functionalities of the system with the respective requirements as detailed in the System Requirement Document (SRD).

Requirement	Functionality
Compute All Pairs Shortest Distance (APSD) in a privacy-preserving manner	The system incorporates privacy-preserving algorithms that allow calculation of APSD between two input graphs without compromising the confidentiality of the graphs.
Compute Single Source Shortest Distance (SSSD) in a privacy-preserving manner	The system incorporates privacy-preserving algorithms to calculate SSSD on two input graphs, ensuring the privacy of the graph data is maintained.
Compute privacy-preserving set union on two input graphs	The system includes an algorithm for privacy-preserving set union, allowing the union of two input graphs to be computed while maintaining privacy.
Handle large graphs in a reasonable amount of time	Efficient algorithms have been used in the system design, enabling it to handle large graphs efficiently.
Ensure the privacy of the input graphs	The system's design ensures the privacy of the input graphs. The data within the graphs is not compromised by the algorithms.
Follow the secure multi-party computation paradigm and the semi-honest model	The system adheres to the secure multi-party computation paradigm and the semi-honest model, ensuring data security throughout.

8. APPENDICES

"Privacy-Preserving Graph Algorithms in the Semi-honest Model" - Original paper introducing the privacy-preserving algorithms implemented in this project.