Zero-Knowledge Proofs for Secure and Private Voting Systems

24-25J-136

Project Proposal Report

Mushtaq M.B.M

BSc (Hons) in Information Technology specialized in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

August 2024

Zero-Knowledge Proofs for Secure and Private Voting Systems

24-25J-136

Project Proposal Report

Mushtaq M.B.M

Supervised by – Mr. Kavinga Yapa Abeywaradana

BSc (Hons) in Information Technology specialized in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

August 2024

Declaration

We declare that this is our own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Mushtaq M.B.M	IT21303920	Mushtaq

The supervisor/s should certify the proposal report with the following declaration.

The above candidate is/are carrying out research for undergraduate dissertation under my supervision.

Supervisor	Date	Signature
K.Y. Abeywardena	22nd of August 2024	[illegible]

Abstract

This paper documents the evolution of a real-world multiparty computation (MPC) protocol to securely, efficiently and verifiably count votes in modern election systems. Nowadays, with election integrity becoming more and more important, not only in the digital era, but this protocol also solves critical security threats using advanced cryptographic tools. To protect the privacy of voters, and to maintain confidentiality over all election data, we make use primarily of an additive secret sharing scheme in which no single party can by themselves reconstruct any secret such as a vote without help from others.

In addition, it uses zero-knowledge proofs (ZKPs) to verify the accuracy of computations without revealing votes. This is crucial to ensure that the election result cannot be changed always enabling the verification of correct results, without violating voter anonymity.

It addresses the problems around scalability and efficiency by adding batch processing tricks into the protocol. The system is able to process large numbers of votes and voters without a significant loss in performance so it could be used for real-world election scenarios. Efforts are also made to perform wide benchmarking which aims at testing the protocol for its various performance metrics like scalability, security or execution time. This combination of features makes the solution solid and ensures that it can be used in secure, transparent elections across any type or scale required in a digital environment.

Contents

Introduction

Today, when we live in a digital society, safeguarding data is a matter of course: we must take every possible step to maintain the security, privacy and integrity of data when computing. This is especially significant in fields such as electronic voting systems. Certainly, traditional forms of voting remain dependable. Even so, they are out of step with the technological world in which we dwell today--problems to be solved include how to prevent fraud, steal people's private information and raise cynicism over the correctness of election results. To this end, cryptography offers a kind of solution. Thus, one of the most promising techniques to come out of this field is Secure Multiparty Computation (MPC).

MPC = it means that many parties computing jointly a function over their inputs and yet those inputs of theirs are kept secret. This is crucial in elections where voter anonymity and the integrity of the vote must be preserved. MPC guarantees that no one party can reconstruct secret information, such as individual votes, alone and unaided by others; hence it is highly relevant for secure voting. The field of secure voting has seen various applications of cryptographic methods to make elections more secure and reliable. Among these, MPC is unique because both privacy and accuracy are preserved.

Earlier studies have shown it can be achieved using additive secret sharing, which breaks down a vote into segments and shares these among multiple people, and Zero-Knowledge Proofs (ZKPs), which prove that calculations have been done right without divulging what the calculations are about--thus preserving vote integrity. In this project, a mastery of techniques such as additive secret sharing and ZEPRKEY PROOFs is needed because they are the basis for secure-vote distribution, integrity validation.

At the same time, techniques of batch processing are essential in processing the enormous number of votes that come in one fell swoop as well. The state-of-the-art in secure voting technology: to address privacy, security and scalability problems involved with electronic voting, advanced cryptographic protocols like MPC are now being tested. Results of these efforts include this project. The task of this project is to build upon previous advancements in this area of inquiry by creating a much more flexible and efficient MPC protocol that relies on additive secret sharing, batch processing and ZKPs. This new protocol aims to remove existing constraints while providing a reliable and scalable one for the politics of today.

Background and Literature Review

Background

To meet these growing requirements for honesty and openness, cryptographic protocols have since been developed. Installing a reliable and secure voting system that allows voters to maintain their privacy and ensures correct election results. Increases in demand for secure, affordable voting systems have fueled research into cryptographic protocols that can maintain the integrity of elections as well as safeguard voter privacy and accuracy in final vote counts and results. In traditional voting systems, both paper-based and electronic, dishonesty and fraud have long been part of the situation. Paper ballots, notwithstanding their physical controllability, have the potential for being tampered with or manipulated in various ways.

They can be changed, destroyed or miscounted easily; and the outcome is the disputes and questions of validity which inevitably accompany suspected irregularities in vote counts. Similarly, electronic voting systems bring about many new security concerns in operation. These potential threats include hacking, unauthorized access of data and software bugs that could compromise the entire voting process. As society becomes increasingly digital, the importance of secure and reliable voting mechanisms becomes more pronounced than ever before. At the same time well, this situation has caused a tremendous surge in exploring advanced cryptographic techniques that can not only ensure security but also transparency and verifiability in elections.

Multiparty computation (MPC) has emerged as one promising answer to the problem. With this technique, our project hopes to provide secure and private (all functions are computed "in the clear") functions for several participants where each party provides part of the data without letting others know what or how much they have contributed.

In the context of voting, this means that votes can be cast, counted, and validated without revealing the choices made by individual voters, thus maintaining privacy protocols tend to use methods such as additive secret sharing, where each vote is divided into several shares that are distributed among the various parties taking part in the protocol. No single party can fully recreate the vote from its own share, thus maintaining the anonymity of voters throughout this entire process. In addition, MPC allows for a decentralized computation of election results; this means that there is no single agency controlling the entire process and greatly reduces risk from tampering or fraud.

MPC provides a solution to the verifiability and transparency which is necessary for elections. By implementing zero-knowledge proofs in MPC one can check that results are always tabulated properly but never present any sensitive data at all. This means to everyone involved that these outcomes can be trusted, making the poll a secure and transparent institution that all can have faith in. Using this combination of cryptographic techniques, voting methods presently in use either offline or via a computer can be overcome. We get new main directions and biking greater vitality out of them than before. We will give you secure voting technology to have full confidence in!

Literature Survey

[1] [3] Multiparty Computation for Secure Voting Systems Desktop Voting in Private Meeting Elsewhere Secure Multiparty Computation (MPC) protocols for voting systems have been extensively investigated, especially emphasizing the need for secrecy and security during vote counting. Efficient MPC techniques that do not depend upon authenticated channels thus providing a stable underpinning for secure collaborative computation in distributed environments. Their work stresses the significance of preserving voter privacy while also assuring that the computational process has not been tampered with.

[2] Scalability and Efficiency in MPC Protocols: One of the main technical problems that arises from applying MPC is its low efficiency and therein lies an architectural challenge to find solutions not only with practicality but also security intact. An arbitrary secure MPC was proposed—a framework that improved the efficiency of MPC protocols. The experiments show us how to reduce computational and communication overhead, so that MPC can be used in actual election cases

[4] Secret Sharing-Based MPC: The use of secret-sharing schemes in MPC Protocols has been a cornerstone of secure voting systems. Escudero (2023) introduces secret-sharing-based MPC, which give a complete overview of how these secret-sharing schemes can be applied to voting systems. This research highlights that secret-sharing is a useful technique for splitting computation among several parties, so that no single party may compromise the integrity of election outcome.

Research Gap

The current MPC protocols provide some degree of security, and privacy is retained. However, they often lag in terms of expandability and speed necessary for large-scale elections with vast numbers of voters or so many people involved as to be literally called astronomical. In a small or controlled environment, these protocols may work fine. But as we multiply their scale, problems manifest themselves. Because we have too many calculations and can't deal conveniently with the increasing volume of information to be processed over ever greater distances in a short time frame--both which increase system slow-down rates-that gives rise to both more resources consumed and lost. Further, ensuring verifiability in such a large-scale setting without sacrificing performance is a major problem. So, there is an urgent need for an MPC protocol robust enough to handle large numbers of participants and votes, and to sustain the highest standards of security, performance and verifiability even in situations as difficult as a general election.

Research Problem

Beyond general accuracy and fairness, cryptographic protocols will soon become necessary for public elections on account of electronic voting. As elections move toward digital systems, it becomes more necessary to have strong cryptographic protocols that can guarantee election outcomes are trustworthy. However, the current MPC protocols, effective though they may be within restricted parameters, are often fraught with limitations when they come to large-scale national elections. These limitations include an inability to efficiently treat many votes at once, the security vulnerability means individual votes could be traced back to their source by unauthorized people, and the leniency of vote counting correctness not only lays claim to privileged information but also full privacy- preserving principle.

In confronting these challenges, the protocol must be designed such that it is scalable by nature. Cap able of processing thousands or even millions of votes without losing efficiency and sophistication. Furthermore, the protocol must incorporate advanced cryptographic techniques to ensure that the entire voting process remains off limits to the outside world-unreadable by anyone who dares try and tamper with it. A highly reliable counting of votes is indispensable. This means that one winner comes out from each game played by these protocols; there can be no forged prizes or broken plays.

Beyond security and performance, the protocol should be easily verifiable as well as implementable in practice. In other words, election officials and independent auditors should be able to confirm the vote count is accurate without seeing any of the voters' underlying private data. The protocol should be user friendly and work with existing election infrastructure to reduce the friction of deploying it. In the end it would be an important pillar for democracy, a solid ground on which future secure voting systems could and will likely be built upon.

Objectives

Main Objectives

Secure and privacy-preserving multi-party computation (MPC): Develop a MPC protocol for voting process of vote counting ensuring the properties such as parties involved are self-explanatory, election should scale to accept 100'000s million voters in final practical solution etc.

Specific Objectives

To ensure a democratic vote counting we need to develop an MPC protocol that is secure, scalable and efficient starting from the design of an algorithm using additive secret sharing. This method of cryptography breaks down the vote into pieces that are handed out among different parties such that no party can peek at only one initiating the original share. This will further maintain voter privacy and prevent any tampering.

Since elections can get BIG, the protocol implements batch processing to manage multiple votes simultaneously. This decreases computational overhead, which improves the scalability of the protocol so that it is applicable to real-world election settings.

Lastly, Zero Knowledge Proofs (ZKPs) are a necessary implementation to prove that computed operation results within the protocol were calculated correctly. In this case, ZKPs enable parties to ensure that the vote counting process is performed correctly without revealing any information regarding who voted for whom (i.e. votes). These techniques in combination—additive secret sharing, batch processing and zero-knowledge proofs make a sound framework for secure and checkable vote counting in elections.

Methodology

1. Vote Collection and Distribution 1.

The process of counting votes commences with the voting of every eligible voter. Separately, each vote is securely sent to the system and broken into many shares through additive secret sharing. These shares are traded through various participating parties so that no one party can get the whole vote. This step is important to keep the voter's privacy and no single entity can dictates or control overall results.

2. Design of Secure Multiparty Computation (MPC) Algorithm

Sending the votes to each other's, with this step we are done and now our solution is ready for building and testing MPC algorithm. This algorithm leverages the properties of additive secret sharing, allowing any number of parties to securely a) count votes b) without seeing actual votes from other party. It should be powerful enough to work across different kinds of elections for a small local election and bigger national level elections.

3. Zero-Knowledge Proofs (ZKPs) for Integrity Verification

In designing the MPC algorithm, Zero-Knowledge Proofs (ZKPs) are included to confirm that each participant can verify that counting is executed exactly as expected without learning any of the votes. Using ZKPs—zero-knowledge proofs—the parties can prove that their computations worked as advertised, so the result i.e. the final vote tally is built on solid and accurate foundations. This is very important to save the purity and transparency of the election process.

4. Scalability: Batch Processing

As existing in some MPC protocols, batch processing is used to deal with a big number of votes individually. By this way, the system will be able to calculate a plurality of votes in bulk at fast pace. The batch processing method is necessary to ensure that the system scales so it can handle elections of epic proportions without any noticeable performance impact.

5. Continuous Monitoring: Security

Real-time security monitoring takes place during the entire vote counting process to find out and handle anything suspicious that can occupy in real life. This system for monitoring is meant to identify any attempt of unauthorized logins or activities, adjusting votes and other malicious activity that would put the recordings in question. Any security concerns are immediately addressed, preserving the continued integrity of vote counting.

6. Final Tally and Verification:

This requires the use of a special kind of designed protocol called secure multi-party computation (MPC) to calculate the final tally after all votes have been processed. These results are then verified using the Zero-Knowledge Proofs to ensure that computational was done correctly and final tally reflects votes accurately. It is essential that Non-compulsory step to prevent the election from being entirely a farce.

7. Reporting and Compliance:

Final tallies are then verified, and results securely transmitted to the appropriate election authorities. The entire vote counting, measures taken by the Company for security and compliance of legal/regulatory requirements are fully documented in this report. It acts as a clear and auditable record for the election process for posterity.

8. Post-Election Evaluation and Benchmarking:

After the election, a post-election analysis is performed to assess how well MPC performs. Performance metrics include scalability, security and execution time which are monitored to evaluate system performance. The conclusions reached in our analysis are used to iteratively revise, refine and ultimately improve the protocol for following elections - thus maintaining it to conform with best practices of efficiency & security.

System Diagram

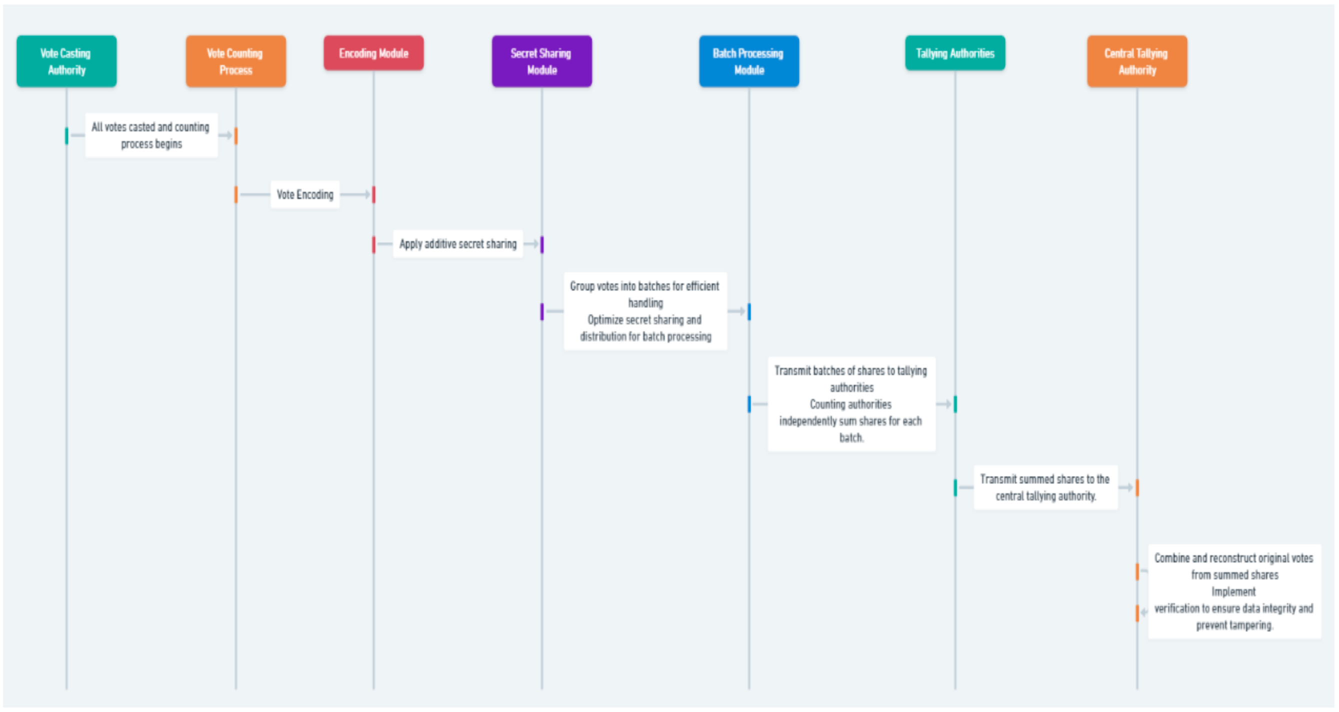
Vote Casting Authority
Vote Counting Process
Encoding Module
Secret Sharing Module
Batch Processing Module
Tallying Authorities
Central Tallying Authority
All votes casted and counting process begins
Vote Encoding
Apply additive secret sharing
Group votes into batches for efficient handling
Optimize secret sharing and distribution for batch processing
Transmit batches of shares to tallying authorities
Counting authorities independently sum shares for each batch.
Transmit summed shares to the central tallying authority.
Combine and reconstruct original votes from summed shares
Implement verification to ensure data integrity and prevent tampering.

Figure No 1: Methodology System Diagram.

Projects Requirements

Functional Requirements

Additive-Secret-Sharing for Secure Vote Sharing

At its heart, the protocol is based on an application of additive secret sharing — a cryptographic method where each vote gets splintered into multiple pieces and shared separately between several entities. While neither party can reconstruct the original vote alone, so that privacy of voters is preserved. In doing so, there's a minimal risk sitting with any one party in displaying or tampering the sensitive voting data. The design of the protocol ensures that only if all parties cooperate will they be able to reconstruct the original vote, taken in full for final tally.

Zero-Knowledge Proofs (ZKPs) For Vote Count Verification

Zero-Knowledge proofs (ZKPs) are used to ensure the integrity and accuracy of vote totals. ZKPs enable one party to prove to another that a computation — such as summing up votes while tallying them together — has been executed correctly without showing any of the original votes. This ensures all the votes are counted correctly and in a transparent manner, while maintaining voters' privacy.

Batch Processing: For Efficient Data Loading of Large Datasets

Since an election could potentially include a very large number of ballots and voters, the protocol uses batch processing methods to improve scalability as well efficiency. The protocol is also scalable with huge number of votes because the system processes multiple votes simultaneously which reduces computation time and resource usage. Thus, the voting can always be done in a timely and practical way even if there is large data being processed.

Non-Functional Requirements

Scalability is an obvious one, as the protocol must support elections where tens of thousands or millions of voters participate. It needs to be able to handle the high load efficiently without affecting its performance or security. This optimization includes minimizing the data processed in each pass and as many operations performed synchronously on a node, allowing it to be efficient as both the dataset size and number of participants scale.

Full security is needed to prevent leaking and tampering of data, or it being misused by other malicious users. The protocol must integrate strong cryptographic protections that will serve both the privacy of voters and hold a proof against any possible doubt on how ballots were counted. Election Guard achieves this by leveraging additive secret sharing to protect the votes, so that even a single party cannot reconstruct the actual secrets without cooperation from other some of all parties ensuring there is no deployed sensitive information.

In elections, results are meant to be computed shortly after voting has closed thus the importance of real-time performance The protocol has to be fast, for it to process and verify the votes rapidly but still without losing accuracy or security (by reading interoperability). When we balance these non-functional requirements, the MPC protocol can deliver a scalable, secure and performant answer for election purposes as they stand today, while meeting stringent demands that are necessary in order have public trust & high degree of reliability.

Expected Test Cases

Several key test scenarios should be considered to guarantee the strictness and trustworthiness of our proposed secure multiparty computation (MPC) protocol for vote tallying. These will be used to validate the protocol for securely sharing and reconstructing votes, verifying computations using ZKPs and batch processing.

The first test case will be to verify the protocol's ability of securely share votes between multiple parties, using secret sharing additive in way that not one party can reconstruct a vote alone. The way the test will work is to take a portion of an election data and redistribute vote shares across multiple parties and then put it back together using only the percentile. The main aims are that no vote can be reconstructed with anything less than the correct number of shares and nothing is lost or corrupt in reconstructing the original votes. This test guarantees the protocol preserves voter privacy and data security throughout voting. In “Verification using ZKP Test” we will verify that the protocol proves correctness of vote counting without revealing actual information about votes. Led by Galois, the test will be centered around a scenario where ZKPs are employed to mathematically prove that votes were correctly tallied without revealing any individual vote boxes. This makes sure that the protocol has nothing but a proof of correctness from this tally yet keeps every vote private. The test will verify the correctness of ZKPs to guarantee accurate, robust vote counting.

The purpose of this test case is to examine the efficiency of batch processing regarding a large amount vote, so we can conclude that it works well. First, the test will emulate a large-scale election with numerous participants and see how quickly and accurately the protocol can run votes in batches. Key Performance Indicators: Processing Time Resource Utilization Correctness of Aggregation Output This test verifies that the protocol can scale as required for high transaction volume voting without loss of performance or security.

These test cases will provide an upper-bounding estimation of the evaluation process for MPC, which is useful to make sure that our actual implementation meets a high standard after integrating all components up to date toward secure and accurate vote-counting in real-world election scenarios.

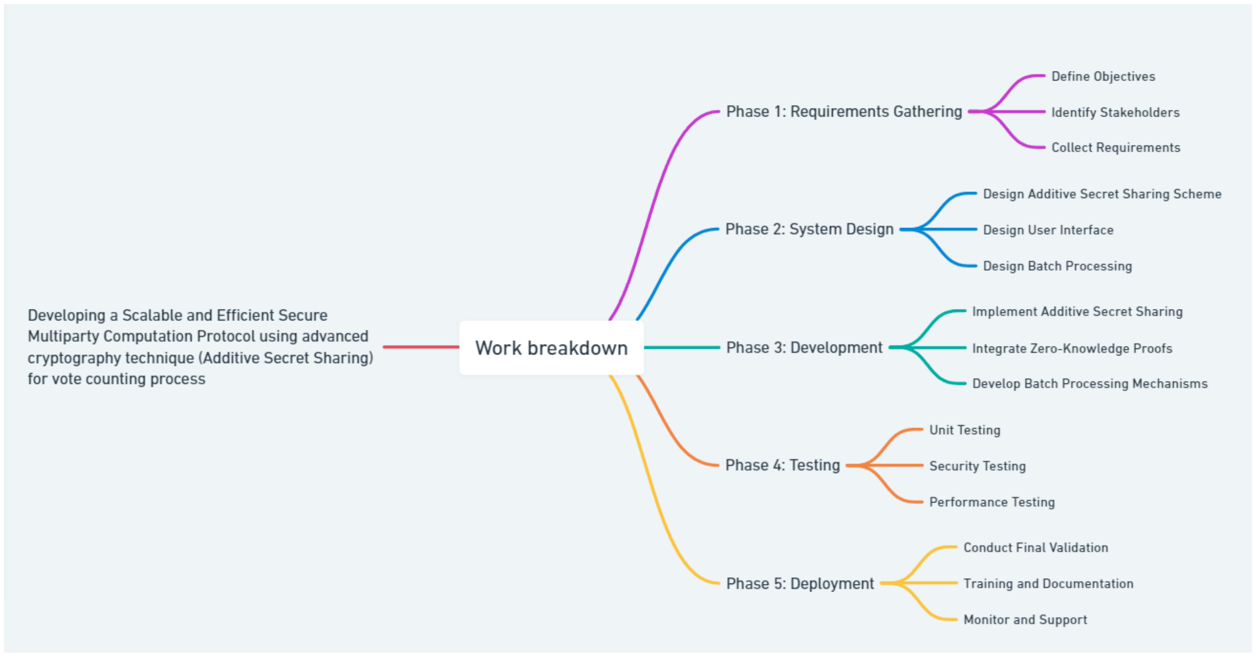
Developing a Scalable and Efficient Secure Multiparty Computation Protocol using advanced cryptography technique (Additive Secret Sharing) for vote counting process
Work breakdown
Phase 1: Requirements Gathering
Define Objectives
Identify Stakeholders
Collect Requirements
Phase 2: System Design
Design Additive Secret Sharing Scheme
Design User Interface
Design Batch Processing
Phase 3: Development
Implement Additive Secret Sharing
Integrate Zero-Knowledge Proofs
Develop Batch Processing Mechanisms
Phase 4: Testing
Unit Testing
Security Testing
Performance Testing
Phase 5: Deployment
Conduct Final Validation
Training and Documentation
Monitor and Support

Figure No 2: Work Breakdown

Budget and Budget Justicfication

The budget for this project has been meticulously planned to ensure cost-effectiveness while achieving the project's primary objectives of developing a secure and efficient multiparty computation (MPC) protocol for voting systems. The emphasis on leveraging existing open-source cryptographic libraries and the university's infrastructure significantly reduces the need for additional funding

Software, Tools, and Programming Languages

- **Open-Source Cryptographic Libraries:**
  - **Sharemind:** A tool for secure multi-party computation.
  - **MP-SPDZ:** A framework for secure multi-party computation that supports various protocols.
  - **FRESCO (Flexible, Efficient, Secure Computation):** A framework designed for efficient and secure multi-party computation.
  - **PySyft:** A library for encrypted, privacy-preserving machine learning.
- **Batch Processing Tools:**
  - **Apache Spark**: A unified analytics engine for large-scale data processing.
  - **Apache Kafka:** A distributed event streaming platform for building real-time data pipelines and streaming applications.
- **Server:**
  - **Microsoft Azure Cloud Services:** Estimated Usage: 300 hours; Cost: $0 (Provided by the university's Azure subscription).

By leveraging these open-source tools and university-provided cloud services, the project effectively minimizes additional costs for the project.

References

[1] T.-D. L. T.-B. H. Duy-Hien Vu, "An efficient approach for secure multi-party computation without authenticated channel," *An efficient approach for secure multi-party computation without authenticated channel,* 2020.

[2] R. (. Cramer, "Secure Multiparty Computation and Secret Sharing," *Secure Multiparty Computation and Secret Sharing,* 2015.

[3] T. S. M. Z. Daniel Demmler, "A Framework for Efficient Mixed-Protocol Secure Two-Party Computation," *A Framework for Efficient Mixed-Protocol Secure Two-Party Computation,* 2015.

[4] D. Escudero, "An Introduction to Secret-Sharing-Based MPC," *An Introduction to Secret-Sharing-Based MPC,* 2023.