Certifiable Consensus using Combined Confidential Computing and Continuous Authentication with Integrity - High-Level Design

Se Elnour[1,2]
[1]Edinburgh Napier University
[2]Blockpass Identity Lab
Supervisor(s): Bill Buchanan

October 23, 2023

1 Introduction

This mini project documentation provides a high-level design (HLD) for a proof of concept (PoC) system that combines confidential computing and continuous authentication with integrity to achieve certifiable consensus. The system leverages various technologies including Microsoft's ElectionGuard Web API, Confidential Consortium Framework (CCF), Intel Sawtooth Lake Distributed Ledger Platform (SLDLP), Hyperledger Cello/Besu with GoQuorum, Hyperledger Indy/AnonCreds, and/or Microsoft Entra Verified DIDs.

2 System Overview

The system is designed to provide a secure, transparent, and verifiable method for conducting certifiable consensus, electronic voting (E-Voting) or internet voting (I-Voting). It uses confidential computing to ensure the privacy and security of the voting process, and continuous authentication to verify the identity of voters.

2.1 Microsoft ElectionGuard Web API

The ElectionGuard API is to be extended *[and implemented using Django or FastAPI, as to be adapted for the below Cello and CCF ingestion]*, for providing the core functionality for the system. It is responsible for ballot encryption, casting, spoiling, and tallying.

2.2 Confidential Consortium Framework (CCF)

The CCF can be used for achieving consensus among the nodes in the system, based on the confidential consortium consensus *[members' votes tallying and automated reporting]* reconciliation PoC *[with the Confidential Consortium Framework (CCF) Data Reconciliation Application]*. It provides a framework for building secure and scalable confidential computing applications.

2.3 Sawtooth Lake Distributed Ledger Platform (SLDLP)

The SLDLP can be integrated into the system to provide an integrity attestation platform for distributed ledger technology (DLT). It includes an SGX-enabled SDK platform, an SGX distributed ledger toolkit, and the Hyperledger Sawtooth DLT; given that Hyperledger Sawtooth supports the PoET-SGX consensus algorithm, which is the first version of Proof of Elapsed Time (PoET) *[also noting that the PoET-SGX relies on a Trusted Execution Environment (TEE), such as Intel Software Guard Extensions (SGX), to implement a leader-election lottery system]*.

2.4 Hyperledger Cello and/or Hyperledger Besu and GoQuorum

Hyperledger Cello (HLC) is used as blockchain provision and operation system *[a restful server's implementation based on Django]*. It provides a multi-tenant chain service efficiently and automatically on top of various infrastructures. Moreover, Hyperledger Besu and GoQuorum are Ethereum clients that are used in the system for creating *[both public and/or private]* permissioned ledger networks and managing smart contracts.

2.5 Hyperledger Indy and/or Hyperledger AnonCreds

Either Hyperledger Indy or AnonCreds can be used for continuous authentication in the system, as the Verifiable Credential (VC) format *[that is a ledger agnostic and with a formal open specification]* for adding important privacy-protecting zero-knowledge proof (ZKP) capabilities to the core VC assurances; while also providing tools for creating, issuing, verifying, and managing verifiable credentials.

2.6 Microsoft Entra Verified DIDs

Microsoft Entra Verified ID can be used as a method for identity verification in the system; to be inter-operable with any Verified (Decentralized IDs) DIDs alternative for Continuous Authentication.

3 Detailed Component Micro-services Design

A low-level design (LLD) is to finalize envisioning each component micro-service of the system that is designed to perform specific functions and interact with other components in a secure and efficient manner.

4 Security Considerations

The system is designed with security as a top priority. It uses confidential computing to ensure that all data is encrypted and secure. Integrity attestation is integrated into the system to verify the integrity of data and operations; while the certifiable consensus is built on the ElectionGuard SDK reference implementation leveraging homomorphic encryption to ensure that votes recorded remain secure and secret; while also allowing verifiable tallying and accurate reconciliation.

5 Performance Considerations

The performance of the system is optimized to handle large volumes of data and high levels of traffic; while also considering to implement an extended Envelope Encryption or Key-Encryption Key (KEK), inside a GPU-accelerated Confidential Computing with Fast Multi-Party Computation (MPC).

6 Future Enhancements

The design of the system allows for future enhancements and scalability.