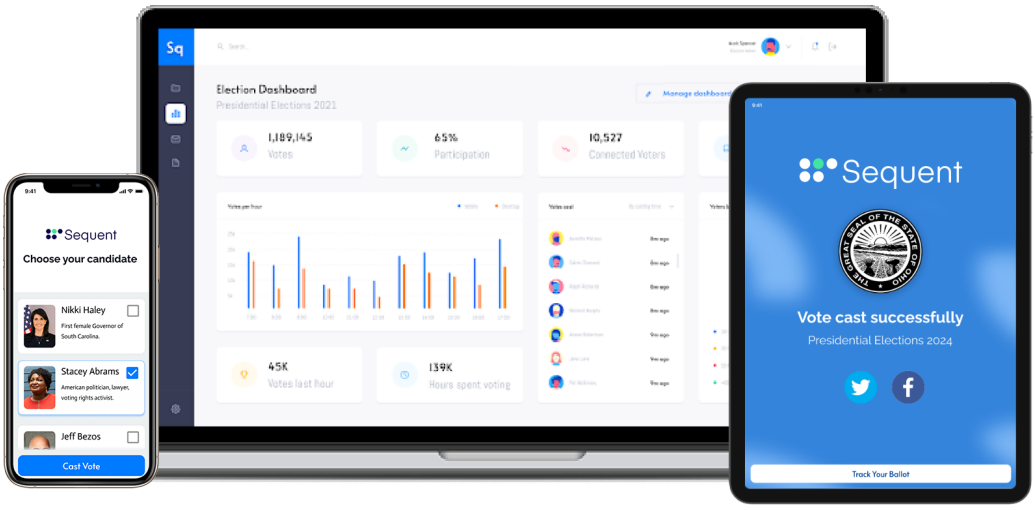

Sequent Tech Cryptographic Protocol

Author: David Ruescas

Document identifier:	2021-09-22-proto-2
Date:	2021-09-22

Introduction

This document provides a high-level description of the cryptographic protocol implemented in the Sequent voting system.

1. Related Systems

The Sequent system employs standard cryptographic techniques following in the steps of well-established voting schemes proposed in the academic literature. Like many other proposed systems, the closest ancestor in Sequent's genealogy tree is Helios[1] in its original mixnet variant. The most significant departures from that Helios design are the use of a threshold distributed key generation mechanism, described in Pedersen[2] and featured in CGS[3] and Distributed Helios[4], and the use of a Terelius-Wikstrom[5] style mixnet rather than the Sako-Kilian[6] one. Other systems with which Sequent shares techniques are Wombat[7] and CHVote[8].

2. The Protocol

Like other end-to-end verifiable voting systems, the Sequent implementation is built to satisfy two main requirements, privacy and integrity.

Privacy is achieved by distributing sensitive information among stakeholders, referred to as trustees, such that *if at least one* of these participants remains honest then voter ballot privacy will be maintained.

Integrity is safeguarded by the verifiability of the operations in the protocol: any observer can check that the outcome of an electoral process is correct. This is achieved without sacrificing privacy by employing zero-knowledge proofs, a cryptographic technique that allows mathematically proving statements without disclosing sensitive information. Using these proofs it is possible to verify that the result of the tally is correct without leaking who voted for what.

2.1. Actors

The most significant participants in the protocol, divided into organizations first and software systems second, are listed below.

2.1.1. Election Authority

The organization or institution wishing to carry out an electoral process.

2.1.2. Voters

The people that the Election Authority has determined are eligible to participate in the electoral process.

2.1.3. Trustee

A person or organization entrusted with sensitive information in order to safeguard ballot privacy. Trustees participate in key steps of the cryptographic protocol.

2.1.4. Authenticator

The backend software component is used to authenticate voters such that they can access the voting booth as an eligible participant (as determined by the Election Authority). Many authentication mechanisms are possible; the (important) details of each of these possibilities are out of scope in this document.

2.1.5. Voting Booth

The software component used by voters to cast their ballots. This software runs on the voter's client device and displays a graphical user interface through which voters make their selections.

2.1.6. Ballot Box

The backend software component used to record and verify ballots (cast through the voting booth).

2.1.7. Mixnet Node

The backend software component used to shuffle (anonymize), decrypt and tally ballots in a distributed manner. Most of the system cryptography is contained in this code. Since Mixnet Nodes handle sensitive information they are administered by Trustees.

2.2. Steps

Following are the key operations of the cryptographic protocol in chronological order, divided in three main phases.

2.2.1. Preparation

The Election Authority:

- Defines the election parameters including contests.
- Specifies the set of eligible voters and the authentication mechanism(s).
- Designates the people/organizations that will act as Trustees.

The Trustees run their Mixnet Nodes to:

- Verify and sign the election parameters.
- Jointly execute the key distribution protocol[2] for each contest in the electoral process, such that each Mixnet Node:
    - Verifies the validity of the contest public key.

    - Holds a portion of the private key material corresponding to the contest public key. The number of such portions required to decrypt ciphertexts is given by the threshold parameter.

- Publish and sign the contest public key for each contest in the electoral process.

2.2.2. Execution

Once the electoral process is open, the Voters:

- Provide credentials to the Authenticator which validates their eligibility. A simple example of an authentication mechanism would be to enter a user password combination on a web page.
- Access the Voting Booth as an authenticated participant and complete their ballot with their choices.
- Optionally spoil the ballot to initiate a ballot auditing protocol[9].
- Cast their ballot (if the ballot was spoilt for auditing it is discarded, only non spoilt ballots may be cast), recording their vote's tracking token displayed by the Voting Booth.

When the Voter casts their ballot, the Voting Booth:

- Encodes user choices as group members of the cryptographic group Gq.
- Encrypts those choices using the corresponding contest public key, generated by the Mixnet Nodes.
- Proves knowledge of the randomness/plaintext to prevent malleability attacks[10].
- Computes and displays a tracking token based on the encrypted vote content, using a one way function.

- Sends the encrypted ballot together with a message authentication code[11] to the Ballot Box through an encrypted channel.

When the vote is received, the Ballot Box:

- Verifies the proof of knowledge of the randomness/plaintext.
- Records the content of the encrypted ballot, optionally including a timestamp and a pseudonym if revoting is required (for example, as a measure of coercion resistance)

2.2.3. Resolution

Once the electoral process is closed, the results are computed. The Ballot Box:

- Extracts the encrypted ballots corresponding to the given electoral process and passes the data to the Mixnet Nodes for processing.

The Trustees run their Mixnet Nodes to execute a mixing protocol[5], for each contest:

- The encrypted votes are re-encrypted and shuffled.
- A proof of valid shuffle is computed.
- The votes and proofs are passed onto the next Mixnet Node, which verifies the previous shuffle, after which the process is repeated starting at the shuffle step.

Once all votes have been shuffled by all the Mixnet Nodes, they initiate a distributed decryption protocol[4] on the outputted votes. The Mixnet Nodes:

- Use their secret key material to compute a partial decryption of each vote together with a proof of correct decryption.
- Verify the other Mixnet Nodes' partial decryptions.
- Aggregate all partial decryptions, thus computing the plaintexts.

- Compute the election results from the plaintexts according to tally specifications defined by the Election Authority.

The Election Authority uses data produced by the Mixnet Nodes to publish the official results and verification data. This marks the end of the electoral process.

2.3. Cryptographic Primitives

We briefly list the most significant cryptographic primitives employed in the protocol. These have been carefully chosen to achieve the required privacy and verifiability guarantees discussed in section 3.

2.3.1. ElGamal encryption over multiplicative groups[12].

Provides ballot privacy modulo the parameters specified in 2.3.3.

2.3.2. Schnorr proof of knowledge of discrete logarithm[14].

Implements proofs of plaintext knowledge at submission time to thwart privacy attacks based on ballot malleability[10].

2.3.3. Pedersen threshold distributed key generation and distributed decryption[13][4].

Distributes secret key material safeguarding ballot privacy as well as providing election integrity through fault tolerance.

2.3.4. Verifiable re-encryption mixnet using Terelius-Wikstrom proof of shuffle[5][16]

Provides universal verifiability through chains of correct shuffle proofs between mixnet node inputs and outputs.

2.3.5. Chaum-Pedersen proof of discrete logarithm equality[15].

Provides universal verifiability through proofs of correct decryption.

3. Privacy and Verifiability

As stated in the introduction the aim of the protocol is to satisfy the requirements of privacy and integrity. How these requirements are met is reviewed here at a high level. Please refer to other documents for more precise details.

3.1. Privacy

The protocol satisfies privacy in the following way:

- The voter's choices are encrypted at the Voting Booth with the election public keys, such that only if all Trustees are corrupt can the ciphertexts be decrypted and the casting voter's choices revealed.
- The votes are only decrypted (for tallying) after being anonymized by the mixnet, such that only if all Trustees are corrupt can the correspondence between input and output ciphertexts be established.

Therefore, privacy is maintained if at least one Trustee is honest. Otherwise nobody can learn the choices an identified voter has made. Furthermore:

- The Authenticator does not reveal voter identities to the Voting Booth, only pseudonyms are transmitted to the Voting Booth, Ballot Box and other systems down the line.

This offers an additional layer of protection.

3.2. Verifiability

The following verifications are possible:

- Voters can verify that their ballots were correctly cast by initiating a ballot audit[9] at the Voting Booth.
- Voters can verify that their ballots were correctly recorded at the Ballot Box using their tracking token.
- Anyone can verify that the set of ballots recorded at the Ballot Box correspond to those outputted by the Mixnet Nodes (proof of shuffle).
- Anyone can verify that the set of plaintexts outputted by the Mixnet Nodes are the result of correctly decrypting the mixed votes (proof of decryption).
- Anyone can verify that the officially announced results have been computed correctly from the plaintexts.

This chain of verifications qualifies the system as end-to-end verifiable.

4. References

[1] Helios: Web-based Open-Audit Voting
[2] Non-interactive and information-theoretic secure verifiable secret sharing
[3] A secure and optimally efficient multi-authority election scheme
[4] Distributed ElGamal à la Pedersen: Application to Helios
[5] Proofs of Restricted Shuffles
[6] Receipt-free mix-type voting scheme — a practical solution to the implementation of a voting booth
[7a] Wombat Voting
[7b] An Implementation of Dual (Paper and Cryptographic) Voting System
[8] CHVote Protocol Specification
[9] Ballot Casting Assurance via Voter-Initiated Poll Station Auditing

[10] How not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios
[11] Keyed-hash message authentication code

[12] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In CRYPTO, 1985.

[13] Torben P. Pedersen. A Threshold Cryptosystem without a Trusted Party (Extended Abstract). In EUROCRYPT, 1991.

[14] SCHNORR, C.-P. Efficient identification and signatures for smart cards. In Advances in Cryptology - CRYPTO '89 (1989), G. Brassard, Ed., vol. 435 of Lecture Notes in Computer Science, Springer, pp. 239–252.

[15] Chaum, David, and Torben Pryds Pedersen. "Wallet databases with observers." Annual international cryptology conference. Springer, Berlin, Heidelberg, 1992.

[16] Verificatum -- An efficient and provably secure mix-net.