Verify-Your-Vote: A Verifiable Blockchain-based Online Voting Protocol

Marwa Chaieb[1], Souheib Yousfi[2], Pascal Lafourcade[3], and Riadh Robbana[2]

[1] LIP2, Faculty of Sciences of Tunis, Tunis, Tunisia,
[2] LIP2, National Institute of Applied Science and Technology, Tunis, Tunisia
[3] LIMOS, University Clermont Auvergne, Clermont-Ferrand, France

Abstract. Blockchain provides the possibility to design new types of applications and systems that allow their users to store data in a secure and transparent way. In this paper, we design a fully verifiable online electronic voting protocol using a blockchain. Our e-voting protocol, called VYV for Verify-Your-Vote, involves cryptographic primitives based on Elliptic-Curve Cryptography (ECC), pairings and Identity Based Encryption (IBE). It ensures the following privacy and security properties: only eligible voter can vote, authentication of the voter, vote privacy, receipt-freeness, fairness, individual and universal verifiability. Furthermore, we formally prove the security of our protocol, using ProVerif tool.

Key words: Online e-voting, Blockchain, Elliptic Curve Cryptography, ProVerif, Verifiability.

1 Introduction

Blockchain [1, 2] is a technology of storage of information. It can be seen as a digital, decentralized, public and large register where all exchanges made between its users are recorded in a public and secure way, without the control of a central entity. It was first introduced in 2008, by Satoshi Nakamoto in his paper [3], where he described a peer to-peer payment system that allows e-cash transactions directly, without relying on financial institutions. In 2014, Vitalik Buterin proposed a Blockchain called *Ethereum* [4]. Ethereum helps us to achieve verifiability, and also to ensure the non-malleability of exchanges in our e-voting systems.

Online e-voting systems aim at providing better level of security than traditional voting systems. Modern cryptography helps us to increase the security comparing to traditional voting systems. We recall the security properties for e-voting systems presented in [5, 6]:

– **Eligibility:** Only the registered voters can vote, and nobody can submit more votes than allowed (typically only one vote per voter is counted, even if several ballots can be casted).
– **Fairness:** No preliminary results that could influence other voters' decisions are made available.
– **Robustness:** The protocol can tolerate a certain number of misbehaving voters.
– **Integrity:** Is the assurance of the accuracy and consistency of votes.

- **Verifiability:** It is usually split into 2 properties: *Individual Verifiability:* Each voter can check whether his vote was counted correctly. *Universal Verifiability:* Anybody can verify that the announced result corresponds to the sum of all votes.
- **Vote-Privacy:** The votes are kept private. This can also be modeled as an unlinkability between the voter and his vote.
- **Receipt-Freeness:** A voter cannot construct a receipt which allows him to prove to a third party that he voted for a certain candidate. This is to prevent vote-buying.
- **Coercion-Resistance:** Even when a voter interacts with a coercer during the entire voting process, the coercer cannot be sure whether he followed his instructions or actually voted for another candidate

In [5, 6] the authors proposed a more fine grain hierarchy for privacy notions. We consider simple versions of such properties and use them to prove the security of our protocol in Proverif. In most of cases, online voting systems that vouch for verifiability, use a trusted web server as a public *bulletin board* to display all public values. We take advantage from Blockchain technology to ensure secure and verifiable elections.

Our Contributions: We design a secure online electronic voting protocol called Verify-Your-Vote (VYV for short). We use Blockchain technology as a bulletin board which allows us to have a verifiable election. The use of Blockchain ensures election integrity thanks to its property of immutability. Moreover, our protocol ensures the following properties: only eligible voter can vote, authentication of the voter, vote privacy, receipt-freeness, fairness, individual and universal verifiability. We also use Proverif in order to formally prove the security of VYV.

2 Related Work

We describe several e-voting systems that claim to provide online elections based on Blockchain, which are used or were supposed to be used for elections in the last years. In Table 1, we summarize the security properties of these voting system.

TIVI [7]: is designed by the company Smartmatic. It is an online voting solution based on biometric authentication. It checks the elector's identity via a selfie. An elector only needs to upload a picture of his face to the system before the vote, and then facial recognition technology compares his facial biometry to the image downloaded during the registration phase. Tivi ensures several security properties such as eligibility since it provides different authentication techniques, votes secrecy thanks to the encryption mechanism and taking advantage from Blockchain technology, universal verifiability and votes integrity are guaranteed. This system provides also voters' anonymity since it includes a mixing phase and individual verifiability by the mean of a QR code stored during voting phase and checked later via a smartphone application. However, this system has some weaknesses. In fact, it does not provide any mechanism to protect voters from coercion. Moreover, voters have the right to vote only once.

Our protocol VYV meets all the properties guaranteed by TIVI. In addition, it provides mechanisms to ensure receipt-freeness and make it possible to an indecisive voter to access to the voting system and change his vote before the end of the voting phase.

Follow My Vote [8]: each voter needs a web cam and a government-issued ID to authenticate himself. A trusted authority verifies the identity of each voter, authorizes only eligible voters to cast their ballots and provides them with pass-phrases needed in case of changing their votes in the future. After voting and casting his ballot to the election Blockchain, each voter is able to see his vote counted in the ballot box. Additionally, voters can watch the election progress in real time as votes are cast. Follow My vote online voting system respects a limited number of security properties. In fact, it includes an authentication phase which ensures voter's eligibility. It allows voters to locate their votes, and check that they are both present and correct using their unique voter ID. Nevertheless, this voting system does not meet several security properties. Indeed, it requires a trusted authority to ensure voter confidentiality and hide the correspondence between the voters' real identity and their voting key. If this authority is corrupted, votes are no longer anonymous. This authority has also the possibility to change votes since it has all voters' pass-phrases so votes integrity is compromised. Votes secrecy is not verified by this system because votes are casted without being encrypted. Moreover, the ability to change votes, coupled with the ability to observe the election in real time compromise fairness property. This system is not coercion resistant and it does not ensure universal verifiability.

Compared to Follow My Vote voting system, Verify-Your-Vote ensures more privacy and security properties. In fact, our protocol provides votes privacy without relying on trusted authorities since it is based on Blockchain and cryptographic primitives (ECC and IBE). We also ensure fairness thanks to the definition of a list of timers that delimit each phase, votes integrity thanks to Blockchain and universal verifiability thanks to ECC.

Open Vote Network [9]: it is a boardroom scale online voting system written as a smart contract on Ethereum. This smart contract is owned by an administrator who is in charge of the election set up and voters authentication. This voting system guarantees diverse security properties. It ensures votes confidentiality since they are encrypted before being cast. It is a self tallying protocol so it warrants universal verifiability. Thanks to the commit phase, it ensures that no partial result can be calculated. Finally, each voter can check that his vote has been recorded as cast and cast as intended by inspecting the Blockchain. Whereas, Open Vote Network is not coercion resistant. It supports only elections with two options (yes or no) and with a maximum of 50 voters due to the mathematical tools that they used. Finally, it needs to trust the election administrator to ensure that only eligible voters have the right to vote.

Unlike this protocol, Verify-Your-Vote protcol ensures eligibility of voters even if the administrator is corrupted because the list of all eligible voters is published and can be verified by everyone. Additionally, our protocol is designed to support large scale elections with multiple options.

Agora voting system [10]: Composed of four technology layers: the Bulletin Board blockchain (based on skipchain architecture [11]), Cotena (a tamper-resistant logging mechanism built on top of the Bitcoin blockchain), the Bitcoin blockchain and Votapp (the application layer of the Agora network). Agora's voting system proceeds in 6 stages. It starts with a configuration phase. During this phase, the election administrator creates a configuration file that contains election parameters. The second step is the casting phase in which voters fill out, review, encrypt and submit their ballots. Ballots are sealed by the Agora's software using the threshold ElGamal cryptosystem. The voter casts her encrypted ballot to the Bulletin Board and signs the transaction with her digital identity credentials. Before casting the vote, a locator (a snapshot of the encrypted ballot) is delivered to the voter. This locator is used for individual verifiability. Next we move to the anonymization phase in which election authority runs all ballots through a mixing network to anonymize the encrypted ballots cast on the Bulletin Board using the Neff shuffling. Once ballots have been anonymized, all the authorities have to collectively decrypt them and publish them with decryption correctness proof. Votes are then calculated. The final result is published on the Bulletin Board. The final step is the auditing phase. Agora's voting system offer the possibility to audit election results at every stage of the voting process. If the election process is successfully verified, a final attestation is signed with the auditors' private key. Our propounded protocol VYV guarantees the same security properties that are respected by Agora voting system. However, we use different cryptographic primitives. In fact, we exploit the Elliptic Curve Cryptography (ECC), pairings and Identity Based Encryption (IBE). An other difference between VYV protocol and Agora voting system is that Agora is a platform that provides, for each stage, different alternatives (for example it supports different modalities of authentication), however our approach is much more precise since it includes specific phases and technologies.

	TIVI	**Follow My Vote**	**Open Vote Network**	**Agora**
Eligibility	✓	✓	Trusted administrator	✓
Fairness	✓	✗	✓	✓
Integrity	✓	✗	✓	✓
Individual verifiability	✓	✓	✓	✓
Universal verifiability	✓	✗	✓	✓
Vote-Privacy	✓	✗	✓	✓
Receipt-freeness	✗	✗	✗	✓
Coercion resistance	✗	✗	✗	✗
Voting policy	Single vote	Multiple vote	Single vote	No available information

Table 1: Security properties of TIVI, Follow My Vote and Open Vote Network.

Outline: In next section, we briefly introduce Ethereum and present an overview of cryptographic primitives used in our protocol. Then in Section 4, we present the structure of the ballots, we also detail the different steps and entities of our protocol. In Section 5, we prove the security of VYV using ProVerif, a verification tool. Finally we conclude in the last section.

3 Preliminraies

This section presents succinctly the basic notions associated to Ethereum and cryptographic primitives used in our approach.

Ethereum: Is a global computer, which anyone can program and use as he wishes. This computer is always on, it is secure, and everything that is done using this computer is public. It allows to develop a new category of applications called decentralized applications [4] (DApps). These applications run on the Ethereum network, which is made up of several thousand computers that constantly communicate. They share the same data which are stored on the Blockchain. Ethereum includes within its blocks executable programs that trigger actions based on information received or conditions reached. We are talking here about smart contracts. A smart contract is an autonomous program that, once started and deployed in the Blockchain, executes predefined conditions. Ethereum can be seen as a transaction based state machine, where transactions can change the state. This latter keeps track of interactions and is composed of ”accounts”. Usually, there are two types of accounts:

- **Externally Owned Account (EOA):** it is a user-controlled account, serves to identify external agents. It is characterized by a public/private key pair. An EOA can send transactions to transfer ether or trigger a contract code. It is controlled by its private key and has no associated code, this private key is used to sign transactions and prove the sender’s identity.
- **Contract account:** is a set of code and data that is located at a definite address. It has only a public key. Its execution is activated by transactions received from other accounts.

Ethereum accounts are identified by their addresses which are constructed from the public key by taking the last 20 bytes.

Elliptic Curve Cryptography: Elliptic curves are used for asymmetric operations such as key exchanges on a non-secure channel, which is called cryptography on elliptic curves or ECC (Elliptic Curve Cryptography). ECC is more efficient than discrete logarithm systems such as DSA [12], ElGamal [13] and also RSA [14]. ECC offers equal security for a far smaller key size.

Pairings [15, 16]: Another advantage of elliptic curve cryptography is that a bilinear operator can be defined between groups. The pairings are obtained from the Weil and the Tate pairing [17, 18] on special kinds of elliptic curves. Let G_1 be an additive cyclic group of order a prime number q and G_2 a multiplicative group of the same order q. A function $e : G_1 \times G_1 \rightarrow G_2$ is called a bilinear cryptographic coupling (also called pairing) if it satisfies the following properties:

1. Bilinearity: for all $P, Q \in G_1$ and $a, b \in Z; e(aP, bQ) = e(P, Q)^{ab}$,
2. Non-degeneration: $e(P, P)$ is a generator of G_2 and so $e(P, P) \neq 1$,
3. Computability: there is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

Identity-Based Encryption [19]: Pairings were introduced into many cryptographic primitives such as signatures or encryption. But the application that remains the most important is Identity-Based Encryption (IBE). Indeed, one of the big problems of public key cryptosystems is the management of keys. There are certification authorities but these are not recognized everywhere and the implementation of a public key infrastructure (PKI) is very expensive. It was Shamir [20], who in 1984 suggested using people's identities as public keys. He proposed to no longer resort to expensive PKI: the key of an individual is directly related to his identity: for example, his e-mail address. The first IBE protocol was proposed in 2001 by Boneh and Franklin [21]. Its main blemish is that the Private Key Generator (PKG) knows the private key. This defect can be corrected by using a distributed PKG like that of Pedersen [22] or Gennaro et al. [23]. In these protocols, a master key is generated in a distributed manner. Each of the m PKGs randomly constructs a fragment. All that is required is that part of these PKGs be online in order to retrieve the private key.

4 Verify-Your-Vote protocol

We design an online e-voting system that uses Blockchain technology, called Verify-Your-Vote (VYV). It is an online voting systems that provides verifiability, uses a *public bulletin board*[1] to display all public values and offers a persistent view to all voters. In VYV, we take advantage from Blockchain technology and explore the feasibility of using this technology as a public bulletin board. We present now the different entities involved in VYV, the ballot structure and the different phases that constitute VYV protocol.

4.1 Protocol entities

Our system is composed of the following entities:

– **Registration Server (RS):** It is a trusted server, its role is to register eligible voters and provide them with their authentication parameters.
– **Election Administrator (A):** It is an externally owned account that manages the election. This includes defining a list of timers to ensure that the election takes place in a timely manner, setting the election parameters, authenticating voters and constructing ballots in cooperation with tallying authority.
– **Eligible Voters (V):** Each voter has an externally owned account. He has the right to vote and casts his choice several times before the end of the voting phase and only his last vote is finally counted.
– **Tallying Authority (TA):** It is an externally owned account. We have as many tallying authorities as candidates. They participate in the construction of ballots, decrypt votes, calculate the election final result and publish the different values that allow voters to check the accuracy of the count.

4.2 Ballot structure of VYV

The main idea of our ballots design is inspired from [24]. We describe the structure of the ballots of VYV in Table 2. Each blank ballot contains four pieces of information: its number "B_N", candidates' names "$name_j$", candidates' pseudo ID "C_j" and the counter-values "C_{Vj}" that are used as a receipt by voters. The ballot number is unique. The

[1] It is a public board where everyone can read and *append only* information. The written data cannot be deleted.

Ballot number B_N			
Pseudo ID "C_j"	**Candidate** "$name_j$"	**Choice**	**Counter-value** "C_{Vj}"
0	Paul	☐	C_{V1}
1	Nico	☐	C_{V2}
2	Joel	☐	C_{V3}

Table 2: Ballot structure.

pseudo ID is the position of the candidate in the ballot, it is calculated from an initial order and an offset value. Counter-values are used for verification. They are calculated for each candidate and depend on the ballot number and the name of the candidate. All these informations are obtained by using cryptography primitives. We detail the construction of ballots in the setup phase.

4.3 Election stage

Here is how the election process works in six steps.
Setup: This phase is described in Figure 1.

1. The election administrator starts by generating the following election parameters and publishes them on the bulletin board:
   - G_1 an additive cyclic group of order a prime number q,
   - G_2 a multiplicative group of the same order q,
   - Hash function: $H_1 : \{0,1\}^* \to G_1$.

   The administrator defines some timers to respect the phases of the election:
   - $T_{beginElection}$**:** The election administrator starts the election process and constructs ballots,
   - $T_{beginVote}$**:** The tallying authority sends ballots to voters. Each voter encrypts his choice and casts it,
   - $T_{finishVote}$**:** Each voter must vote before this time,
   - $T_{beginTally}$**:** The tallying authority decrypts votes and proceeds to the tally,
   - $T_{finishTally}$**:** The tallying authority must finish tallying by this time and begin the verification,
   - $T_{finishElection}$**:** All voters must verify election result before this time.
2. Ballots are constructed in this phase. For each ballot, the election administrator generates a germ g_1 and a random number D_1, encrypts them with his public key PK_A, the obtained cipher is denoted by $\{g_1, D_1\}_{PK_A}$. This value is called the ballot number and is denoted by B_N. He also calculates the offset value using this formula: offset $= H(g_1)$ mod m, where m is candidates' number.
3. The administrator sends the bulletin number B_N and the offset value to the tallying authorities to complete the bulletin construction.
4. Tallying authorities calculate the different counter-values using the following formula: $C_{Vj} = e(Q_{namej}, S_j.Q_{Bn})$, where e() is the pairing function, S_j is the secret key of the tallying authority j, $Q_{namej} = H_1(name_j)$ and $Q_{Bn} = H_1(B_N)$ are two points of the elliptic curve E.

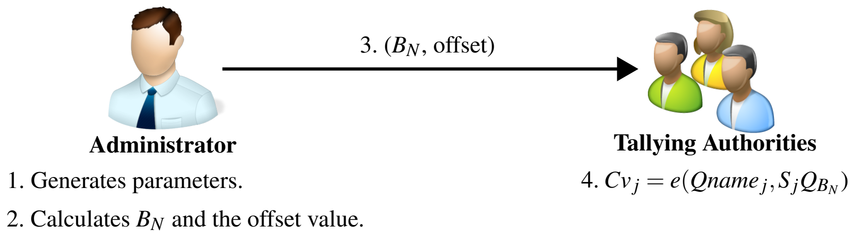
3. (B_N, offset)
Administrator
Tallying Authorities
1. Generates parameters.
2. Calculates B_N and the offset value.
4. $Cv_j = e(Qname_j, S_jQ_{B_N})$

Fig. 1: Setup phase.

Registration phase: This phase takes place offline. The voter resgisters at a polling station. After verifying his legitimacy by a face to face meeting, the voter accesses a server called ”Registration Server”. The registration process is described in Figure 2.

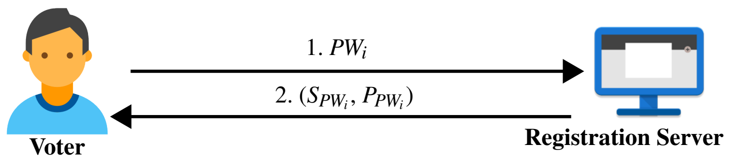

Fig. 2: Registration phase.

1. Every eligible voter enters a password PW_i.
2. The registration server returns to the voter its authentication parameters S_{PWi} and P_{PW_i} where: $S_{PWi} = S_{RS\_A}.H_1(PW_i)$ and $P_{PW_i} = H_1(PW_i)$. Where S_{RS_A} is a secret value shared between the registration server and the administrator.

Authentication phase: We give an overview of this phase in Figure 3.

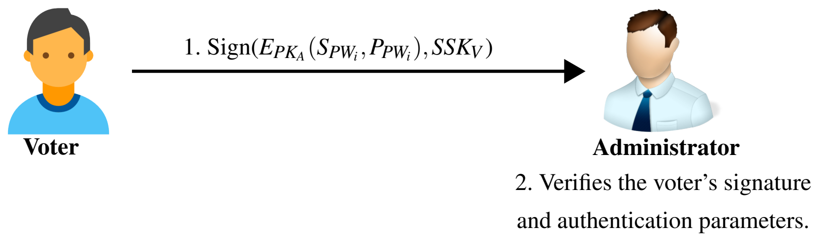
1. Sign($E_{PK_A}(S_{PW_i}, P_{PW_i}), SSK_V$)
Voter
Administrator
2. Verifies the voter’s signature
and authentication parameters.

Fig. 3: Authentication phase.

1. The voter encrypts his authentication parameters (S_{PWi}, P_{PWi}) with the administrator public key "PK_A", signs them with his signature secret key "SSK_V" and sends them to the administrator. To encrypt voter's authentication parameters with PK_A we use Paillier cryptosystem [25], which outlines an asymmetric non-deterministic encryption algorithm with homomorphic additive properties. The security of the Paillier scheme is based on the problem of computing n-th residues in $\mathbb{Z}^*_{n^2}$. To sign voter's authentication parameters we use RSA signature scheme [26].
2. To verify voter's legitimacy, the administrator verifies the voter's signature, decrypts the authentication parameters, calculates $S_{RS\_A}.P_{PWi}$ and compares it with the value of S_{PWi}, if he finds the same value then the voter is registered and has the right to vote.

After being successfully authenticated, each eligible voter has access to an interface that allows him to create his own externally owned account on the election Blockchain and publish his public key as well as his address (which is derived from the public key by taking the last 20 bytes of the hash of the public key).

Voting: This step is illustrated in Figure 4.

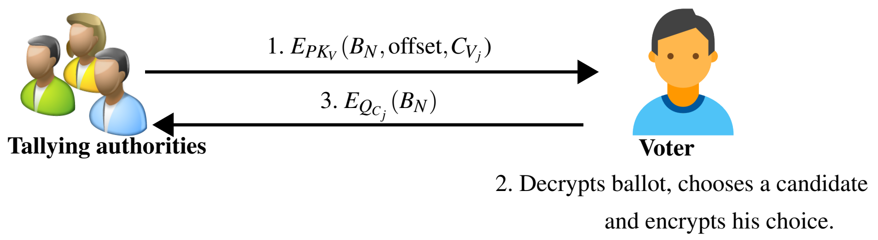
1. $E_{PK_V}(B_N, \text{offset}, C_{V_j})$
3. $E_{Q_{C_j}}(B_N)$
Tallying authorities
Voter
2. Decrypts ballot, chooses a candidate and encrypts his choice.

Fig. 4: Voting phase.

1. The tallying authority randomly chooses a ballot for each voter, encrypts it with the voter's public key PK_v (that has been published on the bulletin board during authentication phase) and sends it to the voter in a transaction on our Blockchain.
2. The voter decrypts his ballot using his secret key, chooses a candidate with pseudo ID C_j and encrypts his bulletin number B_N with $Q_{C_j} = H_1(C_j)$,
3. He casts his vote to the tallying authority via the blockchain.

Each voter should memorize the counter-value corresponding to the chosen candidate and casts its vote before $T_{finishVote}$.

Tallying: Each tallying authority is dedicated to calculate the number of votes of a specific pseudo ID (C_j): for example the first tallying authority "TA_1" decrypts, with its secret key $S_1.Q_{C1}$, all bulletins that was encrypted with the public key Q_{C1} (certainly these ballots contain votes for candidates with $C_j = 0$). Votes decryption reveals numbers of ballots. Each authority consults the exchange history with the administrator stored in the Blockchain to derive the value of the offset corresponding to each bulletin number. From each bulletin number and its corresponding offset, tallying authorities reconstruct ballots, identify chosen candidates and increment counters. The final result of the election must be published before $T_{finishTally}$.

Verification: To verify that the tallying authorities take into account all voters' ballots, the verification phase is organized around two sub-phases: It consists first of all in the reconstruction of counter-values associated with the ballot and the name of the candidate. From the bulletin number and the name of the associated candidate, each tallying authority calculates the corresponding counter-value. Counter-values are published during this phase in the bulletin board. They must be identical to the receipts of the voters. Thus, each voter should access to the Blockchain and check the existence of his counter-value in the list of reconstructed counter-values. The second sub-phase uses the homomorphism property of pairings to check the accuracy of the count. In fact, at the end of tallying phase, each tallying authority "TA_k" publishes on the bulletin board the count of each candidate: $\sigma_{k,name_j} = \sum_{i=1}^{l_j} S_k Q_{B_{N_i}}(name_j)$; Where l_j is the number of votes received by the candidate j, S_k is the private key of the tallying authority k and $B_{Ni}(name_j)$ is the ballot number of the vote i that corresponds to the candidate with name "$name_j$".

Using these values and public counter-values, voters can check on the Blockchain the count of each candidate and the accuracy of the final result. In fact, we have:

$$\prod_{i=1}^{l} C_{V_i} = \prod_{j=1}^{m}\prod_{i=1}^{l_j} C_{V_i}(name_j) = \prod_{k=1}^{m}\prod_{j=1}^{m}\prod_{i=1}^{l_j} e(Q_{name_j}, S_k Q_{B_{N_i}}(name_j))$$

$$= \prod_{k=1}^{m}\prod_{j=1}^{m} e(Q_{name_j}, \sum_{i=1}^{l_j} S_k Q_{B_{N_i}}(name_j)) = \prod_{k=1}^{m}\prod_{j=1}^{m} e(Q_{name_j}, \sigma_{k,name_j}) \quad (1)$$

These equalities use the bilinear property of e:

$$\prod_{i=1}^{l_j} e(Q_{name_j}, S_k Q_{B_{N_i}}(name_j)) = e(Q_{name_j}, \sum_{i=1}^{l_j} S_k Q_{B_{N_i}}(name_j))$$

5 Security of VYV

We first informaly analyze the following security properties for our protocol.

– **Eligible voter:** VYV ensures that only eligible voters join the election Blockchain and participate to the voting process thanks to both registration and authentication phases. During the registration phase, we verify each voter identity via a face to face meeting and only eligible voters are provided with authentication parameters. At the end of this phase, the RS publishes the list of registered voters in the Blockchain. Hence everybody can check the validity of this list. The authentication phase ensures that only registered voters have access to our election Blockchain, create their externally owned account and publish their public keys and addresses. Another advantage of VYV is the linear complexity of the authentication phase. In most cases, online e-voting systems verify the eligibility of each voter by comparing his authentication parameters with a list of registered voters' authentication parameters. Thus, if an election include n voters we have n^2 operations of comparison. Whereas, in our case, we verify for each voter if $P_{PWi} = S_{RS\_A}.S_{PWi}$ so if we have an election with n voters, we execute only n verification operations.

– **Fairness:** Votes are encrypted before being casted so we can not get partial results.
– **Integrity:** The fact of casting and storing votes in the Blockchain safeguard them from being altered or deleted thanks to the immutability property of the Blockchain.
– **Individual verifiability:** This property is respected by our protocol thanks to our ballots structure that includes counter-values C_{Vj}. These values serve as receipts to voters and make it possible to verify that their votes have been cast an intended without disclosing who they voted for.
– **Universal verifiability:** Each tallying authority publishes on the bulletin board the count of each candidate and counter-values. From these parameters, everybody can verify the accuracy of the final result by checking the equation (1). This equation checks the equality between the product of C_{Vj} and the sum of the counts displayed by tallying authorities. If every voter finds his receipt in the list of all counter-values and no claim has been detected and if the equality is verified, then we are sure that the election final result is correct. Thus tallying authorities can not modify, delete or add counter-values since voter's number is public. If one or more tallying authorities try to change the count of one or more candidates, the equation will not be checked and therefore any cheating attempt is detected.
– **Vote-Privacy:** This property is ensured thanks to the use of Blockchain which is characterized by the anonymity of its transactions. In fact every voter is identified in the election Blockchain by a public key and address that have no relationship with his real identity. Votes privacy is ensured even if all authorities present in our approach collaborate. In fact, they cannot establish the relationship between a voter and his vote: the registration server can make the correspondence between the voter's real identity and his authentication parameters, the tallying authorities can make the correspondence between a vote and the public address that sends this vote but no one, except the voter himself, can make the connection between a voter's public address and his authentication parameters since the creation of the Blockchain account is performed by the voter.
– **Receipt-freeness:** In our case, voter cannot find his vote from the counter-value C$_{Vj}$ and the other public parameters. He cannot therefore prove that he voted for a given candidate.
– **Coercion resistance:** Our protocol is not resistant to coercion. A coercer can force a voter to vote for a certain candidate and check his submission later using the counter-value.

Table 3 resumes the security properties of VYV.

Formal verification: We perform an automated security analysis using the verification tool ProVerif [27]. It analyzes secrecy, privacy and authentication properties of

Verify-Your-Vote	
Eligibility	✓
Fairness	✓
Integrity	✓
Individual verifiability	✓
Universal verifiability	✓
Vote-Privacy	✓
Receipt-freeness	✓
Coercion resistance	✗
Voting policy	Multiple

Table 3: Security properties of Verify-Your-Vote.

a given protocol described in Applied Pi Calculus. The Applied Pi calculus is a variant of the Pi Calculus extended with equational theory over terms and functions. It is a language for describing concurrent processes and their interactions on named channels. To describe processes with the Applied Pi calculus, one needs to define a set of names, a set of variables and a signature that consists of the function symbols which will be used in order to define terms. These function symbols have arities and types. In addition, the function symbols come with an equational theory.
To model our protocol in the Applied Pi Calculus, we define a set of types and functions. To represent the encryption, decryption, signature and hash operations, we use the following function symbols: `aenc(x,pkey)`, `adec(x,skey)`, `pk(skey)`, `sign(x,sskey)`, `checksign(x,spkey)`, `spk(sskey)`, `H1(x)`.
Intuitively, `aenc` and `adec` stand respectively for asymmetric encryption and asymmetric decryption, `aenc` and `adec` follow this equation: `adec(aenc(x,y),pk(y))=x`.
The `pk` function generates the corresponding public key of a given secret key. We also assume the hash operation which is denoted with the function `H1`.
The two functions `sign` and `checksign` provide, respectively, the signature of a given message and the verification of the signature. They respect the following equation: `checksign(sign(x,y),spk(y))=x`.

We model our protocol as a single process. All our Proverif codes are avaible on the following website:
`http://sancy.univ-bpclermont.fr/\%7Elafourcade/VYVCodeProVerif`
We also define the following queries to prove votes secrecy, voters' authentication and votes privacy.

– **Verification of votes secrecy:** To capture the value of a given vote, an attacker has to intercept the values of two parameters: the ballot number `Bn` and the pseudo ID of the chosen candidate `Cj`. Thus we use the following queries: `query attacker(Bn)` and `query attacker(C1)`.
When executing the code, ProVerif proves the votes secrecy in few seconds.
– **Verification of voters' authentication:** Authentication is captured using correspondence assertions. The protocol is intended to ensure that the administrator authenticates all voters. Therefore, we define the following events:

  - **event acceptedAuthentication(bitstring, bitstring):** used by the voter to record the fact that it has been successfully authenticated,
  - **event VerifiesParameters(bitstring, bitstring):** used by the administrator to record that he verified voter's authentication parameters.

  We also define the following query:

```
query a: bitstring, b: bitstring;
event acceptedAuth(a,b)==> event VerifyParameters(a,b).
```

  ProVerif proves authentication of voters immediately.
- **Verification of votes privacy:** To express votes privacy we prove the observational equivalence property between two instances of our process that differ only in the choice of votes. To do that, we use `choice[V1,V2]` to represent the terms that differ between the two instances. Likewise, we use the keyword `sync` to express synchronization which help proving equivalences with choice since they allow swapping data between processes at the synchronization points.
  We also succeed to prove votes privacy with Proverif.

6 Conclusion

We have proposed a new Blockchain-based e-voting system. Our propounded solution, VYV, refers to the ECC, IBE and pairings. Through our solution several security properties such as eligibility, fairness, verifiability and receipt-freeness are ensured. Likewise, we modeled the protocol with ProVerif tool and proved that it guarantees votes privacy, secrecy and voters' authentication. Differently from some previous schemes, the recipient of the voter can be used to check his vote while still providing privacy. To provide a higher security level, other properties have to be adopted. Thus, future work will be devoted to guarantee coercion resistance. Finally, the implementation of this protocol constitute interesting future areas of work.

References

1. Aradhya, P.: Distributed ledger visible to all? ready for blockchain? In: Huffington Post. (Apr. 2016)
2. J. Garay, A. Kiayias, G.P.: Proofs of work for blockchain protocols. In: IACR Cryptology ePrint Archive. Volume 2017. (2017) 775
3. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (November 2008) `https://bitcoin.org/bitcoin.pdf`.
4. Buterin, V.: A next generation smart contract & decentralized application platform (2014) `http://www.the-blockchain.com/docs/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf`.
5. Dreier, J., Lafourcade, P., Lakhnech, Y.: A formal taxonomy of privacy in voting protocols. In: Proceedings of IEEE International Conference on Communications, ICC 2012, IEEE (2012) 6710–6715

6. Dreier, J., Lafourcade, P., Lakhnech, Y.: Vote-independence: A powerful privacy notion for voting protocols. In: Foundations and Practice of Security 2011. Volume 6888. (2011) 164–180
7. smartmatic: Tivi. http://www.smartmatic.com/voting/online-voting-tivi/ (2016) [Accessed 11-Dec-2017].
8. Followmyvote: Follow my vote. https://followmyvote.com/ (2012) [Accessed 15-Dec-2017].
9. McCorry, P., Shahandashti, S.F., Hao, F.: A smart contract for boardroom voting with maximum voter privacy. In: Financial Cryptography and Data Security, Springer (2017)
10. Gailly, N., Jovanovic, P., Ford, B., Lukasiewicz, J., Gammar, L.: Agora: Bringing our voting systems into the 21st century. https://agora.vote/Agora_Whitepaper_v0.1.pdf (2018) [Accessed 27-Mar-2018].
11. Nikitin, K., Kogias, E., Jovanovic, P., Gailly, N., L. Gasser, I.K., Cappos, J., Ford, B.: Chainiac: Proactive software-update transparency via collectively signed skipchains and verified builds. In: USENIX Security 17. (2017)
12. National Institute of Standards and Technology: FIPS PUB 186-2: Digital Signature Standard (DSS). National Institute for Standards and Technology, Gaithersburg, MD, USA (January 2000)
13. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Advances in Cryptology. (1985) 10–18
14. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2) (February 1978) 120–126
15. D.Boneh: Pairing-based cryptography: Past, present, and future. In: ASIACRYPT. (2012) 1
16. Rossi, F., Schmid, G.: Identity-based secure group communications using pairings. In: Computer Networks. Volume 89. (2015) 32–43
17. Barreto, P., Kim, H., Lynn, B., Scott, M.: Efficient algorithms for pairing-based cryptosystems. In: CRYPTO. (2002) 354–368
18. Aranha, D., an A. Menezes, E.K., Rodriguez-Henriquez, F.: Parallelizing the weil and tate pairings. In: IMA Int. Conf. (2011) 275–295
19. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: SIAM J.Comput. (2003) 586–615
20. Shamir, A.: Identity-based cryptosystems and signature schems. In: LNCS. Volume 196. (1984) 47–53
21. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: CRYPTO 2001, Springer Berlin Heidelberg (2001) 213–229
22. Pedersen, T.P.: A threshold cryptosystem without a trusted party (extended abstract). In: EUROCRYPT. (1991) 522–526
23. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. In: EUROCRYPT'99. (1999) 295–310
24. Chaum, D., Ryan, P., Schneider, S.: A practical voter-verifiable election scheme. In: ESORICS'15. Volume 3679 of LNCS. (2015)
25. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Advances in Cryptology - EUROCRYPT99. Volume 1592 of Lecture Notes in Computer Science. (1999) 223–238
26. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2) (1978) 120–126
27. Blanchet, B., Smyth, B., Cheval, V., Sylvestre, M.: Proverif 1.98pl1: Automatic cryptographic protocol verifier, user manual and tutorial (2017) http://prosecco.gforge.inria.fr/personal/bblanche/proverif/manual.pdf.