Secret and Verifiable Delegated Voting for Wide Representation

Yefim I. Leifman
rankdemocracy@gmail.com
and the subject must contain the word rankdemocracy
rankdemocracy.blogspot.com

May 15, 2014

Abstract. This paper combines cryptographic voting and web page ranking and proves that it is possible to hold elections so as not to limit a voter by a list of candidates, to benefit from voter's personal experience in dealing with people, to make wide and proportional representation, and to achieve secrecy, including incoercibility, and verifiability of cryptographic voting systems.

Keywords: cryptographic voting, elections, maximum entropy, mixnet, PageRank.

1 Introduction

Progress in the computer field has turned albums into social networks and gave rise to new social phenomena, for example, cryptocurrency and, more generally, crypto-anarchism. These new forms of self-organization of people have been established also with the aim of reducing dependence on governments. Social networks have been used to organize protests leading even to replacement of governments. So, more or less people were killed and state authorities were replaced. Even in the best case there will be elections held in the similar manner as the previous ones and the same people or their opponents will be elected. Can progress in the computer field help to break this circle and to benefit not only the protests against the existing forms of democracy, but to make a contribution to democracy itself? This paper gently explains why the answer is "Yes!"

The first voting by secret ballot was held in Athens under Cleisthenes in the form of ostracism. Let us explore the notion of *voting*. Suppose a voting system has been devised whereby each voter lists all the candidates in order of his preference, and then, by a preassigned rule, the resulting ordering of the candidates is derived

from these lists. All actual election procedures are of this type, although in most the entire list is not required for the choice. Evidently, one wishes that the said rule would have the following properties:

1. If every voter prefers candidate a over candidate b, then so should the output of the voting system.

2. There is no voter i whose preferences are always taken as a result, i.e. there is no dictator.

3. Whether the system outputs candidate a above candidate b depends solely on how the voters ordered candidates a and b, not about where they placed any other candidate.

Let C be a set of candidates, N be a number of voters. Denote the set of all (full and not full) linear orderings of C by $L(C)$. In a full linear ordering the first candidate is preferred above the second etc. In a linear ordering which is not full, neighbor preferences may be indifferent for a voter. Thus a point in $L(C)$ represents a wish of a single voter, a point in $(L(C))^N$ represents a wish of all voters, and a function $F : (L(C))^N \to L(C)$ represents the aforementioned voting rule. Unfortunately, if there are two or more voters that must fully order three or more candidates, then Arrow's Impossibility Theorem [2] states that a function, which corresponds to a rule with the aforementioned properties, does not exist.

The simplest example of a voting system is plurality voting. Plurality voting disobeys Arrow's theorem since voters do not strongly order three or more candidates but prefer up to n candidates above all other candidates. Another example is the *Single Transferable Vote* voting system. Each voter gets one vote, which can transfer from his first-preference to his second-preference etc. If some candidate has more votes than the required quota, he is elected and excluded from the consideration. Also the ballots, where the first preference with his name was removed at the current step, receive reduced weights because one quota of votes was spent. The Single Transferable Vote (STV) is used in Australia. STV results in that fewer votes are wasted. STV obeys Arrow's theorem, indeed, 3 does not hold. Therefore, by the way, STV can be manipulated.

Party-list proportional representation elections can be considered as follows. A voter prefers the first member of his preferable party list over the second etc. and the last over all members of all other party lists of candidates. This voting method disobeys Arrow's theorem since its rule is defined on a restricted domain. Namely, the first member of some party list is always preferred above or indifferent from the second member of the same list etc. According to the Arrow's paper, a voting rule will be said to be *imposed* if for some pair of candidates a and b, candidate a is

always before b in the resulting ordering without any connection with voters' orderings. Therefore party-list proportional representation elections can be considered as imposed in advance. Properties 1 and 3 hold for this method and the imposition takes the role of a dictator.

Advances in *cryptographic voting systems* allow to hold secret and verifiable elections [1], [3], [8]. For example, we want Alice to obtain enough information to personally verify that her vote was correctly recorded, but not so much information that she could convince Carl how she voted. It is not trivial that resolving this conflict is possible! Voting *secrecy* consists of privacy, incoercibility and fairness. *Privacy* ensures that a voter's identity is not linked to the vote he cast. Opportunely, Arrow's theorem applies also to voting without privacy. *Incoercibility* ensures that any coercer, including authorities, cannot force a voter to get the value of his vote, or make a voter to cast votes in a particular way or for a particular candidate. Particularly, a voter cannot prove to a coercer how he voted to sell his vote. *Fairness* ensures that all candidates are given a fair chance by preventing the release of any partial tally such that even counting officials have no clue about partial results. For example, fairness is important for the Single Transferable Vote for which 3 does not hold. *Verifiability* consists of individual verifiability, eligibility, the walk-away property, and public verifiability. *Individual verifiability* ensures that there are mechanisms in place to enable a voter to verify that his vote has been cast correctly and that a voter can file a sound complaint if that is not the case without revealing the contents of the ballot. *Eligibility* ensures that only eligible voters are allowed to vote and prevents the eligible voters from voting twice. *Walk-away property* ensures that valid votes cannot be modified, removed or invalidated from the final tally and if this happens it can be easily detected. *Public verifiability* is the ability for anyone i.e. voters, public auditors, to verify or audit an election to ensure votes have been counted as cast even if several participants colluded. Naturally, all this is possible only if at least a part of voting participants is honest, i.e. behaves according to a protocol.

However, the actual election rules do not exhaust all the ways to rank something. Another way is, for example, the PageRank Citation Ranking which Google uses to rank web pages [5], [9]. PageRank exploits the additional information inherent in the web due to its hyperlinking structure. Thus, link analysis has become the means to ranking. Actually, for pages related to a query, an Information Retrieval score is combined with a PageRank score to determine an overall score, which is then used to rank the retrieved pages. Consider the hyperlink structure of the web as a *directed graph*. The nodes (or vertices) of this directed graph represent the web pages and the directed arcs represent the hyperlinks. (Another example of a directed graph is an underground railway map. Its vertices represent stations and its directed arcs represent a possibility to travel to the next station. The exact

location of the station on the map is irrelevant here, only connectivity matters.) Let a traveler, which in this case is called a *search engine crawler*, walks from one vertex (web page) to another using the hyperlinks which lead to the adjacent vertices (referenced web pages). Assume that the crawler randomly selects an adjacent web page to travel and then travel to it after staying one second on the current web page. In this case, the crawler will soon be stuck on a page without hyperlinks. To prevent this jam we give the crawler a chance to jump from the current page to any page or even to stay another second on the current page. We give the crawler this chance with probability $1 - d$ on each web page, not only on web pages without hyperlinks. This probability is divided equally among all web pages on the web including the web page where the crawler currently resides. The remaining probability d is divided uniformly among chances to travel to the adjacent web pages or joins the aforementioned $1 - d$ if the current page has no hyperlinks. The probability d is called a *PageRank damping factor*. Once it was $d = 0.85$. Assume that one has a huge number of such crawlers, puts each crawler on some web page and they start to travel independently in the aforementioned manner. The theory of Markov chains proves that the fraction of such crawlers residing on each vertex will stabilize over time near some value which depends on a vertex and does not depend on the initial positions of the crawlers. This fraction is the *PageRank* of a web page. Like this a river receives water from confluents (referring web pages) and raindrops (jumps). Thus PageRank is analogous to mean river discharge. Here something like the Water Cycle exists, since each crawler will travel everywhere. One can see that it is desirable that many web pages or web pages with large PageRanks link to a web page in order to the web page had a large PageRank value. Then PageRank will flow to it.

In the actual election rules it does not matter who give you a vote since voting is private for voters. In PageRank it matters who references you. Before Google, search engines used textual content and approximate attendance to return relevant pages. Hyperlink information was wasted. Analogously, in the actual voting rules the better part of information which each person accumulates over her or his life from personal communication with people is wasted. Maybe one can use PageRank to propose election rules which allow to benefit from this wasted information? To my knowledge, a user cannot ask Google to provide pages according to PageRank solely. However, if it were possible, it seems that the retrieved list of the first 100 pages would not reflect the range of the major human interests. Indeed, one needs both an Information Retrieval score and PageRank to build a good search engine. So PageRank per se is not suitable for achieving broad representation because it is too majoritarian. Therefore, in order to design election rules based on web page ranking and to achieve proportional representation, one needs to use PageRank and to require each candidate to state a policy in advance or to design a new appropriate

rank and a rule to elect candidates according to the values of this new rank. We consider the first option as manipulable and proceed with the second.

The appropriate area to search for a suitable pret-a-porter cryptographic voting protocol is liquid democracy. The concept of vote delegation is what all approaches to *liquid democracy* have in common. This means that you as a voter can directly vote on every topic under consideration, but also have the possibility to delegate (proxy) your own vote to a trusted entity. The first example of such cryptographic protocol is described in [11]. This protocol is not suitable for our purposes since, according to the protocol, a vote is anonymized when it is delegated and re-delegation of votes is coercible. Another example is Agora voting. Anybody can create a delegate in Agora, all that is necessary is for the delegate to be appropriately registered in the system. Votes cast by delegates are public, and they must be cast prior to the direct voting period. The Agora direct voting uses a re-encryption mixnet, which we shall explain in Section 5. The authors of Agora voting does not tell anything about cryptography of vote delegation.

This paper positively answers the following question: Is it possible to hold elections so as not to limit a voter by a list of candidates, to benefit from voter's lifelong personal experience in dealing with people, to make wide and proportional representation, and to save the above properties of cryptographic voting systems?

2 Rankdemocracy Rank

We mentioned "quota" when we considered the Single Transferable Vote. For STV the Droop quota is usually applied. The *Droop quota* is $q = \lfloor n/(K+1) \rfloor + 1$, where n is a number of valid votes, K is a number of seats to be filled in the election, $\lfloor\ \rfloor$ denotes rounding down. It is the smallest quota which guarantees that the number of candidates able to reach this quota cannot exceed the number of seats. The Droop quota is also used in party-list proportional representation elections. In party-list proportional representation elections, the number of votes that each party received is divided by the Droop quota, each party receives the number of seats which is equal to the integer part of this quotient and unoccupied seats are allocated according to the fractional parts. There are other quotes and other methods to allocate seats in party-list proportional representation elections. Another method is as follows. Let the i-th party received n_i votes and K_i seats. Let K be the number of all seats and $k_i = K_i/K$. Compute the following sum over all elected parties $S = -\sum_i k_i \log_2(k_i/n_i)$. We suggest to allocate seats K_i so that S would be maximal when n_i are given. In this case voters will be presented with the most uniformity. The quantity k_i/n_i can be considered as a representation of one voter in the parliament. If one assigns a voter of i-th party as a random

dictator with probability k_i/n_i, then S is the *entropy* of the distribution of such random dictators. It is the maximum entropy method to allocate seats. The notion of entropy is motivated by the following facts.

1. The uniform distribution on the interval $[a, b]$ is the maximum entropy distribution among all continuous distributions which are supported in the interval $[a, b]$ (which means that the probability density is 0 outside of the interval).

2. The exponential distribution with mean c is the maximum entropy distribution among all continuous distributions supported in $[0, \infty]$ that have a mean of c.

3. Gas is distributed uniformly in the box in equilibrium and zero gravity.

4. Gas density falls exponentially with height in the Earth's atmosphere.

Therefore the state with maximal entropy corresponds to the maximal "uniformity" under given constraints.

Now we describe *rankdemocracy*. There is no list of candidates. Each voter is also a candidate. Of course, no one will be elected by force. Each voter can vote for up to the predetermined number m of candidates. Say $m = 10$. This number must be justified by means of sociology. Postpone, for a while, how to organize such voting. Consider voting results as a directed graph G. The vertices of G represent voters and the directed arcs represent votes. Denote the set of vertices of G by V. It matters here who gives a vote, but postpone, for a while, loss of privacy. A path p is a $l_p + 1$-tuple of vertices $p = \langle p_0, p_1, ..., p_{l_p} \rangle$ where there is an arc from p_k to p_{k+1} in G and l_p is a length of p. Let $0 \leq d \leq 1$ be the *damping factor* of the rankdemocracy rank analogously to the damping factor of PageRank. We shall describe the rankdemocracy rank formally and after that informally.

Let u_i be the number of arcs which lead out of vertex i. Let $P_{i,j}$ be a set of acyclic (without repeating vertices) paths on G which start at i and end at j. A path $p \in P_{i,j}$ is a $l_p + 1$-tuple of vertices $p = \langle p_0, p_1, ..., p_{l_p} \rangle$ where $p_0 = i$ and $p_{l_p} = j$. Let

$$r_{i,j} = \sum_{p \in P_{i,j}} \prod_{0 \leq k < l_p} \frac{d}{u_{p_k}}.$$

This paragraph is an informal explanation of $r_{i,j}$. During the counting of the votes, the computer does the operations which are equivalent to the following. Voter i puts his autograph on his vote on a sheet of size 1 and keeps it to himself. Further, i can duplicate his own vote, including his autograph, in u_i copies on sheets of size d/u_i and share them with u_i preferred candidates. If voter j gets a sheet of size z with a vote and autographs from other voter and there is no his own autograph on the sheet, then j puts his autograph on the sheet, keeps it to himself, duplicates it

in u_j copies on sheets of size zd/u_j, and share them with those to whom he gave his own sheets. Otherwise j discards the sheet. The sheets collected by each voter are laid in stacks according to the first autograph on each sheet. The total size of the i-th stack of sheets of the j-th voter is $r_{i,j}$.

Let x be a formal variable. Let the polynomial $R_j(x) = \sum_{i \in V} r_{i,j} x^i$ be the *rankdemocracy rank* of vertex j. Let $C \subseteq V$ be a set of elected candidates. Then $\sum_{j \in C} R_j = \sum_{i \in V} W_{C,i} x^i$. Then the value $W_{C,i}$ is the total size of the i-th stacks of all elected candidates. Let $w_{C,i} = W_{C,i} / \sum_{k \in V} W_{C,k}$. The value $w_{C,i}$ can be considered as the representation of voter i in the parliament. We suggest to allocate seats such that $S(C) = -\sum_{i \in V} w_{C,i} \log_2 w_{C,i}$ will be maximal among all permissible subsets C of V. It can be done by Metropolis Monte Carlo sampling using crowdsourcing like bitcoin mining.

It remains to show how to organize such elections and how such elections can benefit from cryptography. We intend to use smart cards to organize such elections. First we describe the most common type of smart cards - *Europay-MasterCard-Visa (EMV) credit card* which is also called "Chip and PIN" credit card. As well this will explain some cryptographic notions on a real example.

3 Cryptography of EMV credit cards

3.1 Cryptographic Algorithms

Let us briefly consider a "Chip and PIN" credit card. Its operation is specified by the EMV Specifications [7]. The EMV Specifications prescribe cryptographic methods to be used for

1. card authentication to a terminal,
2. cardholder (bearer of the card) authentication,
3. secret transmission of data between a card, a terminal, which receives the card, and banks, which participate in the transaction and
4. verification of integrity (inviolability) of data.

The EMV Specifications further describe which cryptographic algorithms are considered reliable. These include:

1. RSA (Rivest-Shamir-Adleman cryptosystem) for the card authentication and cardholder authentication,
2. 3-DES for the data transmission between a card, a terminal and a bank and

3. SHA-1 (Secure Hash Algorithm) for the data integrity verification.

RSA [6] is a realization of public key cryptography. In public key cryptography, each user creates a pair of cryptographic keys - a *public key* and a *private key*. The private key is kept secret, whilst the public key may be distributed to anyone. Messages are encrypted with the recipient's public key and can only be decrypted with the recipient's private key. The keys are related mathematically, but the private key cannot be calculated from the public key in any practical amount of time. Transforming a message with the two RSA keys, public key and private key, successively, in either order, yields the message back.

SHA-1 computes a *secure hash* (or a digest) - a string of fixed length (160 symbols in the case of SHA-1) of zeros and ones, for any given data string (of zeros and ones). The property of the secure hash is that to find a string of data, which corresponds to a predetermined hash, is a practically insoluble task. The combination of the data and its hash, jointly encrypted using a private (secret) key, are commonly referred to as data "*signed*" by this private key. For long data commonly only the hash is encrypted and the signature is the combination of the data and the encrypted hash.

3.2 Card Authentication

Authentication of the information, which is contained on a card, can be carried out by the method of *Static Data Authentication (SDA)*, according to the EMV Specifications. Before a card is issued to a customer - during the process of card *personalization*,

1. the data that identifies the card, such as primary account number (PAN) and expiry date (for the sake of simplicity herein will be referred to as the "card number"), and its hash are encrypted by the RSA algorithm using a private key of the bank and placed on the card;

2. the corresponding public key of the bank and its hash are encrypted by the RSA algorithm using the private key of the credit company and also placed on the card.

The public key of the credit company is available in each terminal. When a cardholder inserts the card in the terminal,

1. the terminal decrypts the public key of the bank and its hash using the public key of the credit company and verifies the integrity of the public key of the bank using its hash.

2. the terminal decrypts the card number and its hash using the public key of the bank and verifies the integrity of the card number according to its hash.

Really, if the card number and its hash correspond to each other, then one who encrypted them knew the private key of the bank. Indeed, the card number and its hash were decrypted using the public key of the bank, whose integrity is similarly confirmed by the signature of the credit company. This method guaranties the authenticity of the information on the card. However, it does not guaranty the authenticity of the card itself. In fact, an illegal card, which contains a copy of the accessible information from a legal card, would pass authentication by this method.

To prevent illegal card duplication, it is necessary that in order to answer questions presented by a terminal, the card would use some information, which cannot be directly read from the card, i.e., the card must encrypt something using its own private key. To this end, the method of *Dynamic Data Authentication (DDA)* is applied. In addition to the card number, the following data is placed on the card during the process of card personalization:

1. the "ICC (integrated circuit card) private key" which will be accessible only to the card itself and cannot be read by the terminal,

2. the corresponding public key of the card, signed by the bank, and

3. the public key of the bank, signed by the credit company.

When a cardholder inserts the card in a terminal,

1. the terminal decrypts the public key of the bank and its hash using the public key of the credit company and verifies the integrity of the public key of the bank using its hash,

2. the terminal decrypts the public key of the card and its hash using the public key of the bank and verifies the integrity of the public key of the card using its hash,

3. the terminal provides an unpredictable number to the card,

4. the card signs the card number and the unpredictable number using its private key. The card then transfers the signed data to the terminal.

5. The terminal decrypts this signature using the public key of the card and verifies the integrity of the card number and the unpredictable number and thus ensures that the card knows its own private key.

Such a card cannot be illegally copied, since its private key, required for this authentication process, cannot be copied. This private key resides in a *tamper-evident secure memory* which must destroy itself when tampered.

3.3 Cardholder Authentication

A "Chip and PIN" card can contain additional public and private keys (called *PIN encipherment keys*) for encryption and decryption of a Personal Identification Number (PIN) using RSA algorithm. Otherwise, public and private keys of the card used for Dynamic Data Authentication can be utilized for encryption and decryption of a PIN. According to the EMV Specifications,

1. A cardholder inserts the card in a terminal and enters his PIN on a *secure tamper-evident PIN pad* to prove his right to use the card.

2. The card generates an unpredictable number and provides it and the PIN encipherment public key to a terminal for PIN encryption.

3. The terminal transfers the public key and the unpredictable number to the PIN pad for encryption of the PIN entered by the cardholder.

4. The PIN pad encrypts the PIN jointly with the unpredictable number and transfers the encrypted PIN and the unpredictable number to the terminal.

5. The terminal transfers the encrypted PIN and the unpredictable number to the card.

6. The card uses the corresponding private key to decrypt the received PIN and the unpredictable number and compares the decrypted PIN and the unpredictable number with the sample being stored secretly in the card.

Then the GENERATE_AC command of the terminal, including Transaction Data (TD), triggers the card to produce a cryptographic signature that can be verified by the bank which issued the card. In particular, if both the card and the terminal agree on completing the transaction offline (based on both entities risk management policies) the card returns a TC (Transaction Certificate) approving the transaction and the terminal sends it to the bank.

4 Cryptography of Voting

In all cryptographic voting protocols a *bulletin board* is made publicly available. Cast encrypted votes are displayed next to voters' names or voters' identification numbers on this bulletin board. One expects enough individual voters to check that their encrypted vote accurately appears on the bulletin board.

If one wants to encrypt a ballot, then one must to pad this ballot with a *secure random* number before encryption. For example, if ballots are either 0 or 1, then

encryption of 2 possibilities using deterministic algorithm gives only 2 ciphertexts and adds only one bit of secrecy. Unpredictable numbers mentioned in Section 3 are also secure random numbers. One says "unpredictable number" when offers the number as a challenge to other party; one says "secure random" (or, when the cryptographic context is obvious, just random) when keeps the number secret. Further, RSA with a public key of length n encrypts a plain text of length n to a ciphertext of length n. An RSA public key of length $n = 2048$ bits (symbols of zeros and ones) is considered now secure [10]. It is equivalent to a decimal number of length 617. Therefore, when one uses RSA with a public key of length n, he must to divide his long plain text to substrings of length n or shorter and to pad each substring. Then each padded substring will be encrypted to a ciphertext of length n. Therefore, if ballots are either 0 or 1 or even can contain one in a billion options, it must be padded anyway.

Then a publicly-verifiable shuffling procedure is run and shuffled decrypted ballots without random padding are published on the bulletin board. Of course, without permuting the results, an adversary would find a voter's preference by comparing the list of voters' names with the list of the plain texts. The shuffling is operated by a trustee. Publicly-verifiable means: trust no one for integrity, but trust the trustee for privacy.

Now the aforementioned trustee knows voters' preferences. To avoid this, Chaum introduced mixnets in 1981. In particular, *decryption mixnet* can be explained as follows. Messages are encrypted under a sequence of public keys using a sequence of random padding values. There are several mix servers. Each mix server shuffles the message order, removes a layer of encryption using its own private key and transmits the results without random padding to the next mix server. In the case of voting these intermediate decryptions without random padding are also posted for the public on the bulletin board. Now the trustees know voters' preferences only if they colluded. The decryptions of the last mix server are plain texts and everyone can tally the election result.

Now two problems remain. The first problem is how a mix server will prove correctness of shuffling and decryption to the public without loss of voters' privacy. The second problem is how to convince a voter that his encrypted ballot is indeed an encryption of his intended plain text vote without revealing the randomization values, that is without loss of incoercibility. Really, the encrypted ballot is posted on the public bulletin board along with the voter's name. Thus, if the voter learns the randomization values, he can prove to a coercer how he voted. We consider only the simplest to explain solutions of these problems.

In 2002, Jakobsson, Juels, and Rivest introduced *Randomized Partial Checking*, a generic mixnet proof system independent of the underlying cryptosystem or shuffling mechanism. This proof system is simple: each mix server reveals an unpredictable

half of its input-output correspondences. A random public draw, such as that used for state lotteries, is performed after the shuffling and ensures that these choices are independent and uniformly distributed. Probabilistically, the mix server cannot cheat more than a handful of voters without to be caught. In more detail, in the first mix server a random half of all correspondences are opened. In the second server the correspondences not pointed to by those opened in the first mix server are opened. In the third mix server another random half of all correspondences are opened, etc. So no complete encrypted ballot to plain text path will be revealed. If at least one such pair of mix servers is completely honest, then the privacy of every input is guaranteed.

Consider the solution of the second problem. Once a voter made his choice, a voting machine encrypts it and displays the ballot plain text and the ballot ciphertext to the voter. Now the voter can audit or seal this ballot or make changes and prepare a new ballot. If the voter choose to audit the ballot, the voting machine reveals the randomization values and a computationally capable voter, equipped with a trusted smart phone, can verify the encryption. Then the voting machine generates a new encryption of the ballot until the voter chooses to seal his ballot. When the voter chooses to seal the ballot, the voting machine discards all randomness and plain text information, leaving only the ciphertext, ready for casting. Probabilistically, the voting machine cannot cheat more than a handful of voters before it gets caught. This method is called *auditing of uncast ballots*. In another version of auditing of uncast ballots, the voting machine displays the ballot plain text and 2 ballot ciphertexts to the voter. The voter must audit one of the ciphertexts and can seal or audit the remaining one. However, since the voting machine does not sign anything, the voter cannot file a sound complaint.

To overcome the aforementioned difficulty, tokenization can be used. A voter choose his preferences and let his trusted smart phone to sign them together using a private key which corresponds to an unpredictable public key. Let the voting machine can read, verify, and save the signature and the public key from the smart phone. Then the voting machine encrypts the voter's preferences and signs the encrypted ballot together with this public key which plays the role of token. Then the voting machine displays the ballot ciphertext, the public key, and their signature to the voter. The voter verifies and saves the signature of the voting machine using the smart phone. Now the voter can audit or seal this ballot or make changes and prepare a new ballot etc. as in the previous paragraph. If the voting machine encrypted an unintended ballot, it cannot show the corresponding voter's signature. On the other hand, if the voter is under coercer's pressure, the voter can show to the coercer everything signed using the same private key. We call this method *auditing of uncast ballots with tokenization*.

The last problem remains: the voter is assumed to believe that the voting ma-

chine discards the ballot plain text and the randomization values. Although the voter does not present himself to the voting machine, the publicly available bulletin board and the correspondence between plain ballots and encrypted ballots can be used to reveal his plain ballot. To overcome this, the voter can encrypt his ballot with several first public keys of the mixnet and then the voting machine completes the encryption using the remaining public keys of the mixnet as in the previous paragraph.

5 Cryptography of Voting using Re-encryption Mixnets

The *ElGamal cryptosystem* is an another realization of public key cryptography. ElGamal with a public key of length n encrypts a plain text of length n to a ciphertext of length $2n$. Similar to RSA, a public key of length $n = 2048$ bits is considered now secure for ElGamal. Similar to RSA, if one uses ElGamal with a public key of length n, he must to divide his long plain text to substrings of length n or shorter and to pad each substring. Then each padded substring will be encrypted to a ciphertext of length $2n$. In the context of voting, in contrast to RSA, one can pad with predetermined padding since there is another way to bring randomness to ElGamal encryption. One must use uniform secure random when encrypts with ElGamal and it is even prohibited to use the same value twice as randomization value with the same key pair. ElGamal offers re-encryption: knowing only a public key, one can re-encrypt a ciphertext to another ciphertext of the same length using a new random. A re-encrypted ciphertext decrypts to the same plain text. Moreover, decryption does not reveal the randomness and the intermediate ciphertext. Another property of ElGamal: a private key can be shared between several trustees and only a quorum of trustees can decrypt ciphertexts. Yet another property of ElGamal: the said private key sharing and public key publication can be achieved without a "dealer" which learns the complete private key.

Aforementioned properties of the ElGamal cryptosystem allow to use it to build cryptographic voting using *re-encryption mixnets* (Sako-Kilian mixnets) instead of decryption mixnets as in Section 4. Initially votes are encrypted using an ElGamal public key and randomization values and published on the public bulletin board. There are several mix servers. Each mix server shuffles the order of votes, re-encrypts them using new randomization values, and transmits the result to the next mix server. These intermediate re-encryptions are also posted for the public on the bulletin board. The output of the last mix server can be decrypted by the quorum of trustees. Once an ElGamal ciphertext is decrypted, this decryption can be proven using the Chaum-Pedersen protocol without revealing the private key and

randomness. Then everyone can tally the election result etc. as in Section 4.

6 Cryptography of Rankdemocracy

Let $Z_K(s_1, ..., s_n)$ denote the joint signature of $s_1, ..., s_n$ using private key K (or the private key which belongs to entity K). The proposed elections consist of the following successive stages.

1. The Central Election Commission gives voters voter's certificates, smart card readers, and trusted operating system live CDs. The voter's certificate has a chip with Dynamic Data Authentication capabilities as explained in Section 3.

2. The Central Election Commission publishes its public key and signed public keys of voting machines and trustees.

3. Several days before elections, the Central Election Commission conducts a public lottery to determine the motto of the elections. Those wishing to participate in the elections sign the motto using the voter's certificate, the smart card reader, and the computer and send the signature to the public bulletin board. The board verifies the signatures and publishes the certificate numbers, the signatures of the motto, the public keys of the certificates, and the signatures of the Central Election Commission under these public keys of those voters whose signatures of the motto was successfully verified.

4. Let A be a set of n trustees. Each trustee A_i has a pair of keys - public key K_i and private key $\underline{K_i}$. Another l trustees share ElGamal private key $\underline{L}$, which corresponds to public key L. Let B be a set of another n trustees.

   When trustee B_0 receives message M, he encrypts the message using public key K_0 with randomization value R_0 and sends the result $M_0 = K_0(M, R_0)$ to B_1. Trustee B_i encrypts M_{i-1} using public key K_i with randomization value R_i and sends the result $M_i = K_i(M_{i-1}, R_i)$ to B_{i+1}. Trustee B_{n-1} encrypts M_{n-2} using public key K_{n-1} with randomization value R_{n-1} and then encrypts the result $M_{n-1} = K_{n-1}(M_{n-2}, R_{n-1})$ using ElGamal public key L with randomization value S_{n-1} to obtain $W_{n-1} = L(M_{n-1}, S_{n-1})$, signs M_{n-2} jointly with W_{n-1} and sends M_{n-2}, W_{n-1} and the signature to B_{n-2}. Trustee B_{i-1} re-encrypts W_i to obtain $W_{i-1} = L(W_i, S_{i-1})$, signs M_{i-2} jointly with W_{i-1} and sends M_{i-2}, W_{i-1} and the signature to B_{i-2}. Trustee B_0 re-encrypts W_1 to obtain $D(M) = W_0 = L(W_1, S_0)$, signs M jointly with $D(M)$ and sends backwards. We shall call such network of trustees a *bilateral encryption network*. The trustees shall endeavor to fight clocked adversaries. There can be several trustees that play the role of B_i, most important, B_{n-1}.

For each voter the board sends voter's certificate number M to B_0, receives $D(M)$ and the signature $Z_{B_0}(M, D(M))$, and puts $D(M)$ and the signature on the board along with M.

5. Trustees, except those from B, can check $D(M)$ for different certificate numbers M by auditing of uncast ballots from Section 4. To initiate such check, the trustee sends the joint signature of M, $D(M)$, and the motto to the board. Each trustee can perform only one series of such tests for each M until another trustee takes the initiative. The board may not replace $D(M)$ on his own initiative.

6. After the trustees, each voter M can do the same for his own $D(M)$.

7. A search utility allows a voter to find the certificate numbers of his candidates. The search utility may even display on demand the driving license format photos of those who wish to take part in the elections. As was mentioned in the Section 1, the voter may vote for no more than a predetermined number of other voters who wish to take part in the elections. The voter reads the bilateral encryptions of the certificate numbers of his candidates from the board to his trusted smart phone, re-encrypts each encryption separately using L and signs the re-encryptions jointly using a private key which corresponds to an unpredictable public key.

   The voter goes to a polling station and comes into a booth. The voter shows his signed re-encrypted preferences to a voting machine using the smart phone. The voting machine reads the preferences and re-encrypts each shown re-encrypted certificate number using L. The voter can check re-encryptions by auditing of uncast ballots with tokenization from Section 4. The voter reads the re-encrypted values, signs them with the private key of his voter certificate, and shows the signature to the machine. After that the voter introduces himself to the machine [4]. The machine checks the signature and reports the results to the local election commission. If the voter used his own certificate, he get the printed receipt with his re-encrypted vote signed by the machine and the machine sends his vote signed by his certificate to the board. The board adds the encrypted information $D(M)$ about the voter M to his encrypted preferences. So each vote contains the encrypted information about the voter and his encrypted preferences.

8. Another set of trustees shuffles the votes using re-encryption mixnet as explained in Section 5. In more detail, for each voter they re-encrypt the information about the voter and about each his preference separately. One can use Randomized Partial Checking to check shuffling as explained in Section 4.

9. A quorum of trustees decrypts the result of the last shuffling using $\underline{L}$ and obtains the graph G from Section 2. This decryption can be proven using the Chaum-Pedersen protocol. The vertices of G are labeled by M_{n-1} values. This labeled graph is published on the board.

10. According to the rank and the maximum entropy rule of Section 2 everyone can compute the vertices which correspond to the elected persons. After that the labels of such vertices can be decrypted to certificate numbers by the trustees from A and the trustees from B reveal the corresponding randomization values. In this case the privacy of elected persons as voters can be compromised if they voted for other elected persons. It is why a voter is allowed to vote for no more than a predetermined number of other voters. Most important, this loss of privacy applies only to a small fraction of voters with big rank values. If someone refused to be elected, one applies the rule of Section 2 again.

11. Trustees reveal randomization values of bilateral encryptions for many non-adjacent vertices of G for public verification.

7 Future work

The main obstacles to the proposed election method are a need for voter education and a need for voter trusted computation. Therefore democracy is a stimulus to close computer backdoors.

Questions remain also in the protocol itself.

- How to determine the maximal number of candidates for that a voter may vote?
- How to determine the damping factor of the rankdemocracy rank? Whether the dumping factor must be determined in advance or must be a function of the obtained graph?
- To which extent the rankdemocracy rank can be manipulated?
- Search for a faster method to determine elected persons according to the maximum entropy rule of Section 2.
- Software implementation.

Maybe someday we will be able to vote for those whom we know.
Updated links to the references are at rankdemocracy.blogspot.com.

References

[1] B. Adida, Advances in Cryptographic Voting Systems. PhD thesis, 2006. http://assets.adida.net/research/phd-thesis.pdf

[2] K.J. Arrow, A difficulty in the concept of social welfare, The Journal of Political Economy, Vol. 58, No. 4. (1950) 328-346. http://cowles.econ.yale.edu/P/cm/m12/m12-03.pdf

[3] V. Augoye, Electronic Voting: An Electronic Voting Scheme using the Secure Payment card System. Technical Report, Information Security Group, Royal Holloway, University of London, 2013. http://www.ma.rhul.ac.uk/tech

[4] J. Benaloh, Ballot casting assurance via voter-initiated poll station auditing. In: Electronic Voting Technology Workshop, USENIX, 2007.

[5] S. Brin, L. Page, The anatomy of a large-scale hypertextual web search engine. Computer Networks 30(1-7) (1998) 107-117. http://www7.scu.edu.au/1921/com1921.htm

[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest, Introduction to Algorithms, The MIT Press, 1990.

[7] EMV Integrated Circuit Card Specifications for Payment Systems, books 1-4, version 4.2, EMVCo, 2008. http://www.emvco.com/specifications.aspx

[8] E-Voting: Can You Trust It? IEEE Security and Privacy, Vol. 2, No. 1, 2004.

[9] A.N. Langville, C.D. Meyer, Deeper inside PageRank, Internet Mathematics Vol. 1, No. 3 (2003) 335-380.

[10] NIST Computer Security Publications - FIPS (Federal Information Processing Standards). http://csrc.nist.gov/publications/PubsFIPS.html

[11] A. Tchorbadjiiski, Liquid Democracy. Diploma Thesis, RWTH Aachen University, March 2012.