Internet voting
Security Requirements, Proposed Techniques and Challenges

1. Introduction

Internet voting is the act of casting a secure and secret ballot over Internet in order to take part and to determine the outcome of an election. [3][11] Internet voting has been widely used in the private sector [1] and recently it has been tested and used, in a limited way, also in the public sector. [4][7] In this paper we would like to focus on the use of Internet voting in the public sector, and in particular on its use during political elections. Political elections are one of the most important and fundamental activities of our society and guaranteeing their correctness and security (in the real world and online) must be a key objective of every civil country. Since the outcome of an election determines the future government and the policy of a nation, the stakes in implementing a correct and safe system are "higher than for many other transactions routinely conducted via the Internet" [1] since "the amount of risk tolerable to financial institutions (even those with billions of dollars of assets) is clearly not acceptable to a country's democratic principles.". [4]

The main topic of this report is to describe which are the requirements of a remote Internet voting system, how it can be implemented, how it has been implemented in the real world, which are the underlying protocols and which vulnerabilities and challenges must still be solved. The structure of the report is as follow. In section 2 we are going to define what are the general requirements of a traditional voting system. In section 3 we will define the concept of Internet voting system and we will briefly present the different types of Internet voting systems. In section 4, starting from the considerations given in the previous paragraph, we will define which are the requirements of a remote Internet voting system. In section 5, we will outline a high-level architecture of a remote Internet voting system implementing the requirements stated before. In section 6 we will show how the high-level architecture has been implemented in the real world for the first time in the US. In section 7, we will focus on the problem of the communication between the client and the server and we will analyse some of the proposed protocols. In section 8, we will briefly list a set of vulnerabilities and challenges that should be considered whenever implementing a remote Internet voting system. In section 9, we will conclude with some consideration about the feasibility of a remote Internet voting system.

2. General requirements of a voting systems

First of all, we are going to examine which should be the requirements of a traditional voting system. We are considering as a reference the well-documented US voting system. In the traditional voting system, on the day of the election, every citizen able to vote visits his local polling place; here, the identity and the right to vote of every voter is verified presenting a valid ID and checking the name on an electoral list; once authenticated, the voter is given a ballot and he is invited to cast his vote inside a voting booth; finally the voter inserts his ballot in a ballot box. [12]

This simple and validated voting system ensures all the following requirements that a democratic election should guarantee: [7]

- ✔ *Authentication*: the identity of the voter must be checked and verified before giving him the possibility to cast his vote. Only people eligible to vote must receive a ballot and must be able to cast their vote;
- ✔ *Ballot secrecy:* the preference expressed by a voter must be kept secret in order to allow a voter to freely express his preferences without being solicited or coerced by anyone and to prevent vote-buying or similar frauds;
- ✔ *Uniqueness of the vote:* a voter must be allowed to cast only a single vote and should be prevented from submitting multiple votes;

- ✔ *Integrity of the vote:* once cast, the preference expressed by a voter must not be modified, forged or deleted;
- ✔ *Reliability of the system*: the polling place must be accessible during the election days and the voter must not be prevented from casting his vote;
- ✔ *Verifiability of the system*: the entire process should be overseeable by any watchdog or auditing organization in order to confirm the right outcome of the election.

In a traditional voting system, all these requirements are easily satisfied by basic physical barriers. [2] The identity of a voter is validated through a valid ID by an election official (*authentication*); the secrecy of the vote is guaranteed allowing the voter to cast his vote inside a polling booth (*ballot secrecy*); the voter can vote only once because before voting his name is marked on the election register *(uniqueness of the vote*)*;* the vote is protected by any tampering because inside the ballot box the vote can not be modified (*integrity of the vote*); the availability of the polling places is guaranteed by the government (*reliability of the system*); the outcome of the election can be verified manually recounting all or part of the votes (*verifiability of the system*).

A particular case of voting is the use of absentee ballot. Absentee ballots are used by voters who are unable or unwilling to cast their vote in the official polling station [7][13]. Usually an absentee ballot is cast via mail (or, sometimes, via a proxy, via electronic mail, via fax or via satellite sites). In the case of absentee ballot voting, *ballot secrecy* requirement results relaxed: in fact, since the place where the vote is cast can not be secured, there is no guarantee that the voter will keep his vote secret, will not be object of external pressures or will not sell it.

3. Types of Internet voting systems

Internet voting is defined as the the act of casting a secure and secret ballot over Internet. [3][11] According to this definition, an Internet voting system can be implemented in many different ways. So, before analysing the particular requirements of an Internet voting systems, we should point out the different possible types of Internet voting systems. [1][2][5][7][8][11] The California Internet Voting Task Force identified four different types of Internet voting systems, which can be considered as different, successive evolutionary stages of the implementation of a remote Internet voting system. [3]

- *Polling place Internet voting:* Internet voting is enabled only in the polling place; every voter must visit his local polling place and there he is given the possibility to cast a traditional paper ballot or to cast an electronic ballot on a dedicated machine;
- *Any polling place Internet voting:* as in the previous case, Internet voting is enabled only in the polling place, but now a voter can cast his vote from any polling place in the county or in the state; he does not need to reach his local polling place any more, but he can use the nearest polling place available;
- *County computer or kiosk computer Internet voting:* Internet voting is enabled from any computer or kiosk owned by the county or the state; the voter is now given the possibility to cast his vote from any machine configured for the election;
- *Remote Internet voting:* Internet voting is enabled from any computer; the voter can now cast his vote from any machine connected to Internet, including his own computer, mobile phone or PDA.

These scenarios are ordered according to the mobility and the freedom given to the voter of voting from any machine. Anyway, as we will see, increasing the degree of freedom means also making it harder to guarantee all the requirements of a voting system. In fact, the number of places from which a voter can cast his vote increases in every stage and enforcing security requirements in every place is often too complex, too expensive or simply impossible. In the following paragraph we are going to take into consideration only a remote Internet voting system.

4. Requirements of a Internet voting system

The first consideration when we think about a remote Internet voting system is that it should

be at least *as secure as* the current traditional paper-based voting system [1][2][9] or as the current absentee ballot system. [3] Starting from the general requirements outlined in the second paragraph, we can now define the requirements of an Internet voting system: [1][2][3][7][9][10][11]

- ✔ *Authentication*: the identity of the voter must be checked and verified before giving him the possibility to cast his vote. Only the votes cast by people who are eligible to vote must be counted and tallied.
- ✔ *Ballot secrecy:* the preference expressed by a voter must be kept secret and a voter must be unable to prove which choice he did in order to prevent any form of coercion or any vote-buying.
- ✔ *Uniqueness of the vote:* a voter must be allowed to cast only a single vote and should be prevented from re-submitting other votes.
- ✔ *Integrity of the vote:* once cast, the preference expressed by a voter must be protected from any tampering on the local machine, on the Internet network, in the remote storage and during the tallying process.
- ✔ *Reliability of the system*: the Internet voting system must work robustly and be always available even in case of machine failures or loss of Internet connection.
- ✔ *Verifiability of the system*: an overwatching organization must be able to verify that all the votes have been counted in the tally and that the outcome of the election is legitimate.

As we said, we listed the requirements of an Internet voting system focusing only on a *remote Internet voting systems*; doing so, we did not reduce the scope of this report: in fact, we can consider all the other types of Internet voting systems as particular cases of the remote Internet voting systems; these particular cases are generally easier to implement because many of the requirements can simply be guaranteed by election officials present in the election venue who can control the access to the voting machines; for example if we consider the *polling place Internet voting,* the *authentication* requirement and the *uniqueness of the vote* requirement do not require any technical implementation and can be simply satisfied by an official in the polling place: the official can manually check the ID of the voters and allow them to cast a single vote.

The list of requirements we produced includes only technical requirements and omits all political, economical and social requirements that a voting system should satisfy (e.g. being accessible to all the citizens, being economically realizable, complying with all the existing laws).

5. A high-level implementation of a remote Internet voting system

Before entering in the details of concrete remote Internet voting system protocols, we want to outline from a high-level how an Internet voting system could and should be implemented in order to satisfy the requirements listed above. We are going to take as a reference the system detailed in [2].

To guarantee that only people eligible to vote are able to do so *(authentication*), we need some form of secure online authentication; the voter must have a way to authenticate himself: for example he could have a <ID, password> pair or he could own a smart card with the relative PIN; the data must then be transferred from the client (voter) to the server (remote voting machine) through a secure encrypted channel.

To ensure the secrecy of the vote (*ballot secrecy*), a voter should be ideally required to vote from a safe and clean machine. When sending the vote through Internet, encryption must be used to prevent the vote from being sniffed. Upon receiving the vote in the remote location the identity of the voter and the preference expressed must be separated so that it would be impossible to connect again the identity of a voter with the vote cast.

To ensure that the voter is able to cast only one vote (*uniqueness of the vote*), the identity of the voter provided through the authentication system must be checked against an electronic electoral register. This requirement can then be guaranteed in two ways: the voter, who has already cast his vote, is prevented from casting another vote; or, the voter is allowed to cast a

vote which will overwrite the previous vote.
To prevent anyone from changing the vote cast (*integrity of the vote*), proper encryption technique must be employed so that the vote can not be tampered by an attacker while travelling over the network.
To ensure that a voter is never prevented from casting his vote (*reliability of the system*) a fail-safe architecture must be designed and properly deployed. Moreover, the Internet voting system must be able to analyse and control the traffic in order to detect and cope with possible denial of service attacks [1].
To guarantee the traceability of the election (*verifiability of the system*) every stage of the election must be verifiable retroactively. A sufficient number of input and output of system must be publicly available to determine if the system behaved correctly or not.

Assuming that the requirements are satisfied and that we can rely on a valid authentication system, on safe voting machines, on electronic electoral registers and on encryption, we can now explain how the system is supposed to work, analysing its behaviour through all the typical steps of an election: registration, validation, collection and tallying. [8][10]
Before the election, the voter applies in the town hall for an electronic ballot in the same way he applies for an absentee ballot; his identity is manually verified by an officer and some days later his credentials for the electronic ballot are sent to his home address.
On the day of the election, the voter sits in front of the clean machine from which he wants to vote; he connects to the Internet voting system and establishes a secure communication channel with an authenticated remote system. On the secure channel, the voter sends the credentials he received and is authenticated by the remote system.
Then the voter is given the possibility to cast his ballot. His vote is encrypted N times and the pair composed by the name of the voter and the vote is sent to an electronic ballot box along with a digital signature to prove the authenticity of the vote.
At the end of the election day, all the submitted pairs are checked against the electronic electoral register; if the name of the voter appears in the register, the name and the vote are permanently separated and the vote is forwarded to one or more centralized tallier; if the name of the voter appears in the register and there is more than one pair belonging to the voter, then only the last vote is forwarded; if the name of the voter does not appear in the register, then his vote is invalid and it is discarded.
Before reaching the talliers, the encrypted votes are submitted to an anonymization network where they are scrambled so that the vote and the origin of the vote can not be linked any more. Every vote go through N different mixers: each mixer (except the last one) can decrypt only a part of the message in an onion-like protocol. Finally, the last mixer produces the votes in plain text and submit them to the vote talliers.
At the end of the election and of the tallying, every mixer publishes all the inputs received and the outputs produced; in this way, anyone can certify the outcome of the election and verify that no fraud was done while transferring votes over the network. If no intermediate result is secret, then the voter can personally follow his vote through the network and be sure that his vote has been counted; however, this last feature, if available, could break the *ballot secrecy* requirement; in fact, if the voter could trace his vote until the last mixer which outputs the decrypted vote, he could prove to anyone how he voted and so he could be able to sell his vote; in other words, this system would be *as secure as* an absentee ballot voting system but not as a traditional voting system.

6. A real-world implementation of a remote Internet voting system

Using the system detailed in the previous paragraph as a model, we can now show how a real-world remote Internet voting system can be deployed. We are going to illustrate the system implemented in Arizona by the Arizona Democratic Party and *election.com* for the presidential preference primary on the 7^{th} – 10^{th} March 2000. The details of this system are given in [4].

In January 2000, the Arizona Democratic Party sent to all the 849.000 registered Democrats in the state a first class sealed mail containing a unique seven-digit alphanumeric personal

identification number (PIN); the number was randomly generated by *election.com* and it was assigned to the voters in a non-sequential fashion, such that, given a PIN, it was impossible to guess the next logical PIN (this measure was implemented to avoid that voters sharing the same physical street address could determine the PIN of the other voter).
On the day of the election, voters connected to the Democratic Party Party website or to the *election.com* website; they received a certificate from the remote server and, once validated, they established a SSL connection to the remote site. The voter was then asked to authenticate himself on the secure connection; he presented his PIN and then confirmed his identity answering two personal challenge questions (e.g.: date of birth or social security number) randomly selected among five possible questions.
After this step the voter could cast his ballot. On his local machine, the Java applet responsible for the election encrypted the ID of the voter and his vote using the public key of a third-party tallier, in this case KPMG's key, which was generated using Certicom's elliptic curve algorithm. The pair ID-vote was then sent to *election.com*. Without KPMG's private key, *election.com* could not decrypt the information received and so it simply stored the encrypted ID and the encrypted vote; to ensure non-linkability between the voter and his vote, the ID and the vote were separated and stored in two different tables inside an encrypted relational database; the access to the database was strictly controlled (possible only through tested stored procedures) and audited.
At this time, once the vote was cast, the voter's PIN was voided in order to prevent him from voting multiple times.
Finally, when the election was over, the votes (and only the votes) stored in *election.com* database were copied to a Zip disk and manually given to KPMG. KPMG decrypted the votes using its private key; without the knowledge of the voter, stored in *election.com* database, KPMG could not connect a vote to a voter in any way. After the successful decryption, KPMG tallied the vote and published the result of the election.
The source code behind this architecture was controlled through commercial source-code management software which tracked all the changes.
The application and the databases were all located at undisclosed site (to reduce the danger of physical attacks) and the access to any of these sites were controlled by card-based and biometric (fingerprint) access control system. The sites were provided with power backup systems and every component in the architecture had from one to seven failover component. All the data stored on the databases were replicated to a stand-by server to guarantee the highest safety of data.
Even if it is impossible to be completely immune to denial-of-service, *election.com* implemented an architecture including intrusion detection systems able to detect anomalous traffic and to configure firewalls and external routers in order to filter and minimize the effect of a possible denial-of-service.
The result of the five days of election was, from a technical point of view, successful: the system behaved properly and was available for 95 hours out of 96; the only period during which it was non reachable was during the first hour of March 7th when a hardware failure in a router (solved in one hour) prevented voters from casting their vote.

7. Protocols for an Internet voting system

We are now going to analyse the protocols that can be implemented to build a remote Internet system. In the following pages we will use these conventions:

- V_i: voter i;
- K_i: symmetric key for V_i;
- AC: authentication centre, validator centre or remote election server;
- PK_{AC}: authentication centre i's public key;
- SK_{AC}: authentication centre i's private key;
- TC *or* TC_i: tallier i;
- PK_{TCi}: tallier i's public key;
- SK_{TCi}: tallier i's private key;

- *B( ):* blind-ing function

7.1 Basic protocol

This is the first and easiest protocol designed for a remote Internet voting system. [8] It is also the protocol underlying the model described in the previous paragraph.

(1) Before the election, every V_i receives a unique ID;
(2) Every V_i submits to the *AC* the authenticated packet {ID, PK_{TCi}{vote}};
(3) *AC* verifies the identity of V_i and checks him off on the election register;
(4) *AC* forwards PK_{TCi}{vote} to *TC*;
(5) *TC* decrypts PK_{TCi}{vote} using its private key and adds the vote to the tally.

This simple protocol is easy and flexible. According to the requirements we defined, it guarantees *authentication*, *uniqueness of the vote* and *integrity of the vote*; it can guarantee *ballot secrecy* if and only if the validator and the tallier does not collude (if they team up the validator can reveal its private key to the tallier and allows it to link every voter to his vote); it can not guarantee the *verifiability of the system* since the voter can not trace his vote in any way; finally, the *reliability of the system* depends on the implementation and can not be evaluated from this protocol.

7.2 Two agency protocol

The *two agency protocol* was proposed by Nurmi, Salomaa and Santean in 1991 to solve some of the problems of the *basic protocol*. [8]

(1) Before the election, *AC* sends to each V_i a secret validation tag;
(2) *AC* sends to *TC* the list of all validation tags with no information on the corresponding voter;
(3) On the election day, V_i sends to *TC* the packet <validation_tag, K_i{validation_tag, vote}>;
(4) *TC* verifies the validation tag inside the packet, but can not decrypt the vote;
(5) *TC* publishes K_i{validation_tag, vote};
(6) V_i checks if his vote has been correctly published;
(7) When the election is over, V_i sends K_i to *TC*;
(8) *TC* uses K_i to decrypt and to tally the vote;
(9) *TC* publishes the pairs <K_i{validation_tag, vote}, vote>;
(10) V_i checks if his vote has been correctly tallied.

This alternative protocol fixes one of the drawbacks of the *basic protocol:* traceability; it guarantees *authentication*, *uniqueness of the vote*, *integrity of the vote* and *verifiability of the system* (every voter can check the list published by the tallier and verify whether his vote has been received and correctly counted); it can guarantee *ballot secrecy* if and only if the validator and the tallier does not collude (if they team up the validator can reveal the association between a voter and a validation tag to the tallier and allow it to link every voter to his vote); finally the *reliability of the system* depends on the implementation and can not be evaluated from this protocol. It should be also noted that the tallier can compromise the election casting a ballot for all the voters who received a validation tag but did not cast their ballot.

7.3 One agency protocol

The *one agency protocol* was conceived by Salomaa in 1991; it is a slight variation of the *two agency protocol*, in which, to avoid the problem of the collusion between the validator and the tallier, the validator is removed and the tag distribution is performed by the tallier. [8]

(1) Before the election, *TC* sends to each V_i a secret validation tag;
(2) On the election day, V_i sends to *TC* the packet <validation_tag, K_i{validation_tag, vote}>;
(3) *TC* verifies the validation tag inside the packet, but can not decrypt the vote;
(4) *TC* publishes K_i{validation_tag, vote};
(5) V_i checks if his vote has been correctly published;
(6) When the election is over, V_i sends K_i to *TC*;

(7) *TC* uses K_i to decrypt and to tally the vote;
(8) *TC* publishes the pairs $<K_i\{\text{validation\_tag, vote}\}, \text{vote}>$;
(9) V_i checks if his vote has been correctly tallied.

The distribution of the validation tags is done using an *all-or-nothing disclosure of secrets protocol*; such a protocol guarantees that every voter can receive one and only one validation tag and that the tallier can not know which validation tag a user received.
The *one agency protocol* can guarantee *authentication*, *ballot secrecy*, *uniqueness of the vote*, *integrity of the vote* and *verifiability of the system*; the *reliability of the system* depends on the implementation and can not be evaluated from this protocol. As the previous protocol, also this protocol has the flaw that the election could be compromised if the tallier casts a ballot for all the voters who had received a validation tag but did not cast their ballot.

7.4 Blind signature protocol

One of the most practical voting protocol using blind signatures is the one introduced by Fujioka, Okamoto and Ohta in 1993. It was developed to solve the problem of the collusion between the validator and the tallier present in the *two agency protocol*. [8]

(1) Before the election, every V_i is recorded along with his keys;
(2) Every V_i creates a voted ballot, encrypt it with a secret key K_i and blinds it: $B(K_i\{\text{vote}\})$;
(3) Every V_i signs the blinded $B(K_i\{\text{vote}\})$: $<B(K_i\{\text{vote}\}), \text{signature}>$;
(4) Every V_i sends the packet $<B(K_i\{\text{vote}\}), \text{signature}>$ to *AC*;
(5) *AC* verifies if the signature of the packet belongs to a V_i who has not yet cast his vote;
(6) *AC* signs $B(K_i\{\text{vote}\})$: $<B(K_i\{\text{vote}\}), SK_{AC}\{B(K_i\{\text{vote}\})\}>$;
(7) *AC* sends $<B(K_i\{\text{vote}\}), SK_{AC}\{B(K_i\{\text{vote}\})\}>$ to V_i;
(8) V_i removes the blinding layer: $<K_i\{\text{vote}\}, SK_{AC}\{K_i\{\text{vote}\}\}>$;
(9) V_i sends $<K_i\{\text{vote}\}, SK_{AC}\{K_i\{\text{vote}\}\}>$ to *TC*;
(10) *TC* verifies the signature $SK_{AC}\{K_i\{\text{vote}\}\}$ using PK_{AC};
(11) *TC* publishes $K_i\{\text{vote}\}$;
(12) V_i checks if his vote has been correctly published;
(13) When the election is over, V_i sends K_i to *TC*;
(14) *TC* uses K_i to decrypt and to tally the vote;
(15) *TC* publishes the pairs $<K_i\{\text{vote}\}, \text{vote}>$;
(16) V_i checks if his vote has been correctly tallied.

This protocol relies on the use of a special type of digital signatures called *blind signatures*; when a blind signatures is placed on a document, it is possible to sign the document without revealing the content. In this voting protocol, a blind signature is placed on the encrypted vote sent by a voter to a validator; thanks to it, a validator can sign the encrypted vote without the need of removing the blind signature; then, because of the properties of a blind signature, the voter can remove the blind signature and can obtain the encrypted vote with the signature of the validator.
The *blind signature protocol* can guarantee *authentication*, *ballot secrecy*, *uniqueness of the vote*, *integrity of the vote* and *verifiability of the system*; the *reliability of the system* depends on the implementation and can not be evaluated from this protocol. With this protocol, not the tallier, but the validator, could cast fake votes for all the voters who did not cast their ballot.

7.5 Sensus protocol

The *Sensus protocol* was implemented by Cranor and Cytron in 1996. It is based on the *blind signature protocol*. [8]

(1) Before the election, every V_i is recorded along with his keys;
(2) Every V_i creates a voted ballot, encrypt it with a secret key K_i and blinds it: $B(K_i\{\text{vote}\})$;
(3) Every V_i signs the blinded $B(K_i\{\text{vote}\})$: $<B(K_i\{\text{vote}\}), \text{signature}>$;
(4) Every V_i sends the packet $<B(K_i\{\text{vote}\}), \text{signature}>$ to *AC*;
(5) *AC* verifies if the signature of the packet belongs to a V_i who has not yet cast his vote
(6) *AC* signs $B(K_i\{\text{vote}\})$: $<B(K_i\{\text{vote}\}), SK_{AC}\{B(K_i\{\text{vote}\})\}>$;

(7) AC sends $<B(K_i\{\text{vote}\}), SK_{AC}\{B(K_i\{\text{vote}\})\}>$ to V_i;
(8) V_i removes the blinding layer: $<K_i\{\text{vote}\}, SK_{AC}\{K_i\{\text{vote}\}\}>$;
(9) V_i sends $<K_i\{\text{vote}\}, SK_{AC}\{K_i\{\text{vote}\}\}>$ to TC;
(10) TC verifies the signature $SK_{AC}\{K_i\{\text{vote}\}\}$ using PK_{AC};
(11) TC sends to V_i $<K_i\{\text{vote}\}, SK_{TC}\{K_i\{\text{vote}\}\}>$;
(12) V_i verifies the signature $SK_{TC}\{K_i\{\text{vote}\}\}$ using PK_{TC};
(13) V_i sends K_i to TC;
(14) TC uses K_i to decrypt and to tally the vote;
(15) When the election is over, TC publishes the pairs $<K_i\{\text{vote}\}, \text{vote}>$;
(16) V_i checks if his vote has been correctly tallied.

The *Sensus protocol* was designed as a replacement for the postal mail balloting systems; the main difference from the *blind signature protocol* is the use of a receipt $<K_i\{\text{vote}\}, SK_{TC}\{K_i\{\text{vote}\}\}>$ from the tallier; this receipt allows the voter to immediately check the signature of the tallier and send his key without waiting for the publication of the encrypted vote $K_i\{\text{vote}\}$ as in the *blind signature protocol*. [10]
The *Sensus protocol* can guarantee *authentication*, *ballot secrecy*, *uniqueness of the vote*, *integrity of the vote* and *verifiability of the system*; the *reliability of the system* depends on the implementation and can not be evaluated from this protocol. Being an implementation of the *blind signature protocol*, also this protocol suffers from the flaw that the validator can cast fake votes for all the voters who did not cast their ballot.

7.6 CJC protocol

The *CJC protocol* was designed by Chen Y., Jan and Chen C. in 2004. This protocol relies on RSA encryption. [9]

(1) Before the election every V_i is recorded and receives a personal certificate C;
(2) TC_1 and TC_2 publishes a number N (product of two primes) and their common public key $PK_{TC1+TC2}$;
(3) V_i sends to AC $<C, v_i>$, where v_i is a pseudonym that will be used by V_i;
(4) AC verifies if the personal certificate C is valid and has not been used before;
(5) AC signs v_i: $<v_i, SK_{AC}\{v_i\}>$;
(6) AC sends $<v_i, SK_{AC}\{v_i\}>$ to V_i;
(7) V_i encrypts his ballot as $b = (K_i \oplus \text{vote})^{(PK_{TC1+TC2})} \bmod N$;
(8) V_i sends to TC_1 and TC_2 the packet $<v_i, SK_{AC}\{v_i\}, b, K_i>$;
(9) TC_1 and TC_2 verifies the signature $SK_{AC}\{v_i\}$ using $PK_{AC}\{v_i\}$;
(10) When the election is over, TC_1 and TC_2 produce the secret key $SK_{TC1+TC2}$;
(11) Under the supervision of TC_2, TC_1 decrypts the votes and publishes them.

In the *CJC protocol*, TC_1 and TC_2 are supposed to be two different entities: a tallying centre and a supervision centre responsible for overwatching the tallier; they share a common public key $PK_{TC1+TC2}$ but the corresponding secret key, $SK_{TC1+TC2}$, is not known at the beginning and will be computed at the end of the election using the secrets held by TC_1 and TC_2.
The *CJC protocol* can guarantee *authentication*, *uniqueness of the vote* and *verifiability of the system*; since $<v_i, SK_{AC}\{v_i\}>$ is not encrypted, *integrity of the vote* could be compromised if an attacker is able to intercept a packet $<v_i, SK_{AC}\{v_i\}, b, K_i>$ and alter it substituting b with b' and K_i with K_i'; the *ballot secrecy* could be violated using a brute force attack: since the marked ballots are limited, an attacker can produce all the ciphertexts using K_i, $PK_{TC1+TC2}$ and N and compare them with b; the *reliability of the system* depends on the implementation and can not be evaluated from this protocol.
A modified and improved version of the *CJC protocol*, using ElGamal cryptosystem is presented in [9].

7.7 Full codesheet protocol

The *full codesheet protocol* was designed as a protocol able to minimize the trust put in the

voter's PC. [11]

(1) Before the election, every V_i receives his personalized codesheet;
(2) On the day of the election, V_i authenticates with AC;
(3) V_i looks up in the codesheet the alphanumeric code c corresponding to his choice;
(4) V_i sends c to TC;
(5) TC sends to V_i a verification code v;
(6) V_i verifies on his codesheet whether v correspond to his choice;
(7) When the election is over, the votes are tallied by TC.

The basic idea behind the *full codesheet protocol* is the use of personalized codesheets; every codesheet contains two random alphanumeric strings for each possible choice; the first string is the code that the voter has to submit to cast his vote; the second string is a validation code which will be returned by the tallier; if the returned code and the string on the codesheet match, than the vote has been correctly received by the tallier.
The *full codesheet protocol* can guarantee *authentication*, *ballot secrecy, uniqueness of the vote, integrity of the vote* and *verifiability of the system*; the *reliability of the system* depends on the implementation and can not be evaluated from this protocol. In this protocol, the *integrity of the vote* can be guaranteed even on compromised machines because the codesheets are always out of the reach of the voter's PC (they are sent by mail and no computation on the voter's PC is required) and so no malicious software can realistically guess the personal values on the codesheet of a given voter.
Easier version of this protocol are the *code number-only protocol* (which does not use any validation code) and the *verification number-only protocol* (which uses only validation codes and which can guarantee a reduced level of *ballot secrecy*). [11]
The use of codesheets can also be paired with the use of *test ballots*. Test ballots can be considered as a type of IDS designed for remote Internet voting systems: special test ballots are periodically sent from the voter's PC to the tallier; the tallier verifies the ballots and if it finds out that these ballots are generated in a statistically meaningful way, it can detect an attack. [11]
Another protocol based on the use of codesheet is presented in [5].

8. Vulnerability of an Internet voting system

8.1 Vulnerabilities at physical level

Independently by any protocol flaws, one of the first basic vulnerability identified in a remote Internet voting system is its vulnerability to pure physic attacks. [2] The requirement of *reliability of the system* can be easily compromised by disruption in the telecommunication infrastructure or by the destruction of power supplies. Any network congestion or network outage can compromise the outcome of an election. [7]

8.2 Vulnerabilities on the client machine

Another very hard-to-solve challenge resides in the remoteness of the voter's machine. The environment in which the voter cast his vote can not be realistically secured. The environment includes the place, the network and the computer where the vote is cast.
An insecure place could compromise the *ballot secrecy* requirement since the vote could be seen by a familiar (if the vote is cast at home) or it could be recorded by close-circuit camera (it the vote is cast from the workplace). [7]
An insecure network could be managed by a network administrator who has the right to access the computer of the voter; in this way the administrator could be able to see or even to tamper the vote cast by a voter; on the other hand, even a secure and properly configured network, in which a firewall block the outgoing traffic, could prevent the voter from casting his vote. [3]
An insecure computer, in which the operating system or the browser have been compromised, can not guarantee a secret and unmodifiable vote; viruses or trojan horses targeting the voting system could easily propagate from computer to computer and undermine the entire validity of

the election. [3][4] The valuable extensibility and flexibility of modern PCs and softwares allow an attacker to easily compromise the security of the voter's machine. [11] There is a countless number of malicious programs that can be executed on a host; these programs, operating before any authentication and any encryption is applied to the vote, can see and manipulate a vote; they can compromise a machine in a rough way (modifying the BIOS, making the machine unbootable and preventing a voter from casting a ballot) or in more subtle way (accessing the browser configuration and inserting as a proxy the machine of an attacker); moreover they can be coded to be very hard to detect (stealth), to operate only on precise days or on targeted machines (trigger), or to delete themselves after the execution so to be untraceable. [6] All these programs can reach a target machine via physical installation (e.g.: disks), via remote automated delivery (e.g.: mail worms), via a web browser applet (e.g.: ActiveX), via installation of a unknown software (e.g.: a software containing a hidden program or a software installing a dynamically linked library or overwriting an operating system modules) or even via installation of a known software (e.g.: when installing well-known software product, even if the software vendor may have no interest in compromising an election, a rogue software programmer could have added code to subvert the voting system). [6]

8.3 Vulnerabilities at network level

At network level, even if the cryptography technology is mature and extremely reliable and the public-key infrastructure is sufficiently developed, [6][7] one of the main vulnerability is constituted by the threat of massive distributed denial-of-service; as now, no effective defence can protect an Internet voting system from a well-planned distributed denial-of-service. [4][6][7]

Another vulnerability of the architecture, resting on the inability of most of the user to distinguish between a legitimate server and a non-SSL connection, is DNS poisoning; a malicious user could spoof a legitimate server sending fake mails or targeting a Domain Name Service in order to redirect all the voters to his website; if the user is unable to understand SSL-generated warnings, the attacker could pretend to be the original election website or could mount a man-in-the-middle attack. [6]

9. Conclusions

In January 2000, the shared[6] and often-quoted [1][4] conclusion of the California Internet Voting Task Force was that “it is technologically possible to utilize the Internet to develop an additional method of voting that would be at least as secure from vote-tampering as the current absentee ballot process in California. At this time, it would not be legally, practically or fiscally feasible to develop a comprehensive remote Internet voting system that would completely replace the current paper process used for voter registration, voting, and the collection of initiative, referendum and recall petition signatures”. [3]

Further development in the hardware and computer design could lead to platform able to create a *trusted path* between the client and the server; the Trusted Computing Platform Alliance (TCPA) or the Extremely Reliable Operating System (EROS) are addressing the problem of creating platform safe from any interfere by malicious program. [11]

As now, the weakest ring of the chain in a remote Internet voting system is the voter's computer. “Excluding the expensive solutions (trusted hardware) and the idealistic ones (clean operating systems)” [5], excluding closed secure devices [11] and the use of an approach *security through obscurity* [11], the most promising solution seems to be the use of codesheet and test ballots. In fact, a *trusted path* over Internet between the voter and a remote server can be easily created using reliable technologies like a certificate infrastructure and SSL/TLS [11], but it is very hard or nearly impossible to build a *trusted path* between the voter and his machine.

Only accepting that a remote Internet voting system can be *as secure as* the absentee voting system and not as a standard voting system and recognizing that the voter must hide his vote to his own untrusted PC, it could be possible to develop a safe remote Internet voting system.

REFERENCES:

[1] Coleman K., *CRS Report for Congress – Internet Voting,* Jan 2003 (last update); publicly available at:
http://www.ipmall.info/hosted_resources/crs/RS20639.pdf
(last access: 04 April 2008)

[2] Election Technique 2000 Commission, *Excerpts from Technology and Administration in Election Procedure*, 2000; publicly available at:
http://www.governments-online.org/documents/InternetVotingSweden.pdf
(last access: 04 April 2008)

[3] California Internet Voting Task Force, *A Report on the Feasibility of Internet Voting,* Jan 2000;
http://www.sos.ca.gov/executive/ivote/final_report.pdf
(last access: 04 April 2008)

[4] Mohen J., Glidden J., *The Case for Internet Voting*, January 2001;
Phillips D.M., Von Spakovsky H.A., *Gauging the Risks of Internet Elections,*Jan 2001; publicly available at:
http://www.21cconsultancy.com/Julia%20ACM%20Article.pdf
(last access: 04 April 2008)

[5] Nitschke L., *Secure Remote Voting Using Paper Ballots,* Apr 2008;
publicly available at:
http://arxiv.org/PS_cache/arxiv/pdf/0804/0804.2349v1.pdf
(last access: 04 April 2008)

[6] Rubin A.D., *Security Considerations for Remote Electronic Voting over the Internet,* Dec 2002; publicly available at:
http://avirubin.com/e-voting.security.pdf
(last access: 04 April 2008)

[7] Internet Policy Institute, *Report of the National Workshop on Internet Voting: Issues and Research Agenda*, Mar 2001; publicly available at:
http://news.findlaw.com/cnn/docs/voting/nsfe-voterprt.pdf
(last access: 04 April 2008)

[8] Cranor L.F., *Electronic Voting – Computerized polls may save money, protect privacy*, 1996; publicly available at:
http://www.acm.org/crossroads/xrds2-4/voting.html
(last access: 04 April 2008)

[9] Dou B., Chen C., Araujo R., *Attacks and Modifications of CJC's E-voting Scheme,* 2006; publicly available at:
http://eprint.iacr.org/2006/300.pdf
(last access: 04 April 2008)

[10] Cranor L.F., Cytron R.K., *Design and Implementation of a Practical Security-Conscious Electronic Polling System*, Jan 1996; publicly available at:
http://www.cs.wustl.edu/cs/techreports/1996/wucs-96-02.ps.Z
(last access: 04 April 2008)

[11] Oppliger R., *How to Address the Secure Platform Problem for Remote Internet Voting,* Oct 2002; publicly available at:
http://www.ifi.uzh.ch/~oppliger/Docs/sis_2002.pdf
(last access: 04 April 2008)

[12] Wikipedia, *Voting;*
publicly available at:
http://en.wikipedia.org/wiki/Voting
(last access: 04 April 2008)

[13] Wikipedia, *Absentee Ballot;*
publicly available at:
http://en.wikipedia.org/wiki/Absentee_ballot
(last access: 04 April 2008)