Blind Signatures and Blind Signature E-Voting Protocols

Michael Schmid and Andreas Grünert

University of Applied Science Biel, 2502 Biel/Bienne, Switzerland

Abstract. The purpose of this report is to summarize and explain blind signatures, explain e-voting basics with emphasis on privacy and exemplify how blind signatures are used in e-voting schemes. This report does not claim to contain new findings by the authors, but is meant as a summary and introduction on the subject. As many reports on blind signature e-voting protocols presented are not easily understandable without having a master degree in mathematics and the audience is not such, we intend this report to be understandable with a basic knowledge of cryptographic methods. Because of this, some problems or other properties in protocols mentioned are not scientifically proven here. Such proof can, however, be found in the papers referenced throughout the text.

1 Introduction

1.1 Structure

This document is structured in two main parts. Section 2 on page 3 explains the origin, the definition, classes and usages of blind signatures independently from e-voting systems. Section 3 on page 7 then introduces e-voting and explains how blind signatures are used within e-voting schemes. In that part, we have a special focus on privacy within e-voting schemes (since blind signatures are a means to achieve privacy) and go into some detail on the subject how e-voting schemes generally are faced with a dilemma between balancing out receipt-freeness and verifiability.

1.2 Terms, Protocols and Schemes

In this report we not only write about blind-signatures but also mention a few other topics. To have a thorough understanding of the text, we explain the most important ones briefly:

E-Voting: The term "e-voting" generally refers to *remote* electronic voting, meaning voting over an insecure network, for example the internet. The remoteness-property is however not necessarily part of all e-voting systems. For instance, electronic polls are systems that replace the ballot box with an electronic device, without delocating the voting booth. In this text we use e-voting to express remote electronic voting only. Other terms attached to voting are survey, poll and election, which we use interchangeably.

Bit Commitment Scheme: Bit commitment defines a scheme where someone makes a commitment, but hides it from the public until he decides to open it. Compareable to a last will that is only opened after the death of the author. In such a scheme, the author symmetrically encrypts his commitment with a secret key that the receipient of the commitment does not know, and hands it to the recipient. As soon as the commitment is supposed to be opened, the author gives the recipient the key. This is necessary to ensure that the author can not change his commitment in the meantime. A special type of this scheme called *trap door bit commitment*. It allows the author to use a trap door key instead of the real one for opening. The commitment seems valid, but the use of the trap door key can be recognized. See [17] page 40ff for a detailed explanation.

Anonymous Channels: Anonymous channels are used to anonymize a message sent from one peer to another, such that the origin of the message can not be traced. There are several types of implementations known: Mix-Nets, Onion Routing, Re-Encrypting Networks and more. For an extensive explanation see [17] page 20ff.

Zero Knowledge Proof: A zero knowledge proof refers to a proof system where the existence of a relation or an element is not proven by revealing it, but by a system that enables the verifier to deduce this assumption from other information.

2 What Blind Signatures are

Blind Signatures are a well-known method to preserve privacy and anonymity within complex transactions. These two properties are given by the method as follows:

Privacy: To have a message signed by a trusted authority without this authority knowing about the content of the message. Next to the blinded document it is necessary to send proof of eligibility to the trusted authority.

Anonymity: For a third party to know that the eligibility of the originator to release that message is checked by a trusted authority, without knowing anything about the originator.

Blind signatures were introduced by David Chaum in [1] (1981) to realize an untraceable payment system by cryptographic methods (see Fig. 1 for a simplified overview of the protocol). His aim was to have a digital-money based payment system in which the payee let the bank sign a cheque to be handed to a third party. The system must provide an inability of the third party to determine any information about the payee, but retaining proof of payment by the payee. *Chaum* was mainly interested in the anonymity property. Since the bank has to know how much money the payee wants to debit from his account and therefore sign, the payee has to tell this next to the blinded request. The privacy property was defined as ”desired property”, but not used for the proposed ”untraceable payment system”.

2.1 How blind signatures work

The main feature that defines blind signature schemes is that the signature on the blinded document also signs the unblinded document. In other words, it must be possible to remove the ”blindness” factor without invalidating the signature. All proposed signature schemes build upon public key cryptography. We will focus on the first published one from David Chaum [2] that bases on the RSA scheme. It is explaned in Sect. 2.2.

2.2 RSA blind signature scheme

The RSA blind signature scheme is quite straight-forward. It is implemented with the same parameters and elements as the RSA scheme used for encryption, decryption or signing, except for a blindness random value. The following explanation is taken mainly from [17] page 17ff and [23].

Simply put, there are three parties to this action: The author of the message, the signer (who confirms eligibility of the author by signing the message), and a third party that can verify the signature and therefore the eligibility of the message. Each participant has an RSA public key K and private key K^{-1} (a

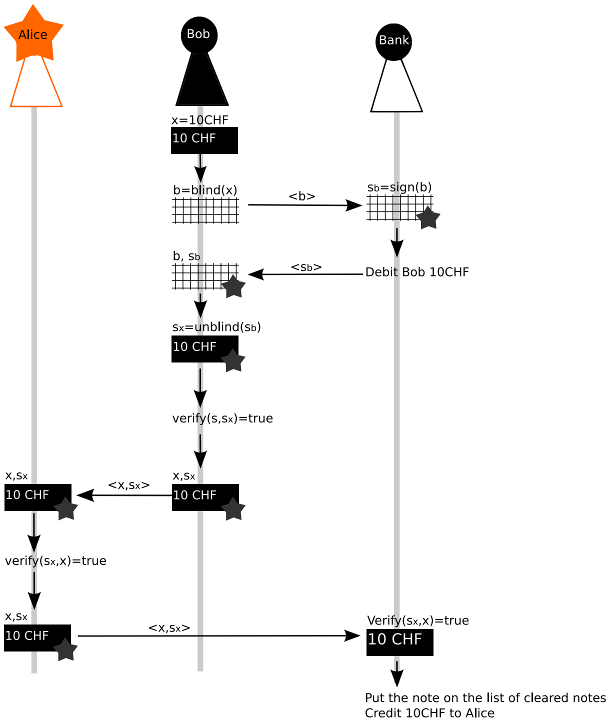

Fig. 1. The idea is that Bob can give Alice money without the Bank knowing about it. First Bob blinds his request to debit a defined amount of money from his account. The money itself is then "stored" in the digital note (x in the illustration), guaranteed by the Banks signature (s_x). Then Bob transmit this note to Alice - anonymously or directly. She passes the note back to the Bank to be credited on her account. Since s_x can not be deduced from s_b (which the Bank could trace back to Bobs request), the Bank cannot track the payment.

key consists of a modulus n generated from the multiplication of two large prime numbers, an exponent and the key values K or K^{-1} respectively).

When the author wants to blind sign a message m he has to go through six steps:

I) The author choses a secret random number r whose greatest common divisor with n is 1.

$$gcd(r, n) = 1 \qquad or \qquad r \bmod n \equiv 1$$

II) He now encrypts r with the public key of the signer and multiplies it with the message.

$$x \equiv r^{K_{\text{signer}}} \cdot m \pmod{n}$$

III) The value x represents the *blinded message*. This blinded message is now sent to the signer.

IV) The signer signs the message with his private key. The signer itself cannot know what m is without knowing r.

$$t \equiv x^{K^{-1}_{\text{signer}}} \pmod{n}$$

V) t is now a valid signature on the blinded message. The signer returns this value to the author.

VI) The author can now retrieve the signature on m easily by multiplying the signature t on the blinded message x with r^{-1}. This works because:

$$r^{-1}\cdot t \equiv r^{-1}\cdot x^{K^{-1}_{\text{signer}}} \equiv r^{-1}\cdot (r^{K_{\text{signer}}}\cdot m)^{K^{-1}_{\text{signer}}} \equiv r^{-1}\cdot r\cdot m^{K^{-1}_{\text{signer}}} \equiv m^{K^{-1}_{\text{signer}}}$$

This is very neat since the author got the valid signature from the signer, as though the signer had signed the original message directly!

2.3 Privacy Classification of Blind Signatures

In the definition above, the security of the protocols is considered only at the moment of signing the document. But it is also an issue if the signature keeps its anonymity when the unblinded, signed message is shown to the signing authority again at a later point in time. This aspect leads to a classification of *privacy types*. The classification is according to Horster/ Peterson from [6] and partly opposing the classification given by Franklin/ Yung in [5].

Hidden Signature: The signer can recognize the unblinded message by comparing the signature with the one he stored together with the initial eligibility proof and blinded message. He therefore knows the content of the message after the signed unblinded message has been released to public.

Weak Blind Signature: The signer can recognize the unblinded message even though the signature of the blinded message is different to the unblinded signature, because there exists a relation between them.

Strong Blind Signature: Even if the signer sees the unblinded message with the signature he can make no conclusion about the original blinded request. This is the class David Chaum originally proposed in [1]. The RSA blind signature scheme is a strong blind signature scheme.

2.4 Uses of blind signatures

Digital Cash: Digital cash was the reason why David Chaum 1982 proposed blind signatures (see [1]), since they allow a mechanism to have trust without traceability by third parties.

E-Voting: In 1992 the first widely accepted e-voting protocol was proposed by A. Fujioka et al. in [4]. To deal with the anonymity and privacy properties, they included blind signatures in the first phase, the voters registration. This report is mainly about this use of blind signatures. More details can be found in Sect. 3.

Accredited Pseudonyms: To know that you speak to a eligible person without actually knowing who it is.

E-Notary: To notarize digital contracts, which must be kept confidential.

3 How Blind Signatures are Used Within E-Voting Protocols

3.1 Voting Security Requirements

The following list gives an overview of requirements needed for voting to be secure and democratic. The list is largely equivalent to the one given in [17], page 11. Blind signatures are only used in e-voting protocols to establish *privacy*.

Unconditional Requirements

- **Completeness:** All valid votes are counted correctly.
- **Soundness:** Invalid votes should not be counted.
- **Privacy:** All votes must be kept secret.
- **Unreusability:** No voter can vote twice.
- **Eligibility:** Only those who are allowed to vote can vote.
- **Fairness:** Nothing must affect the voting, no intermediary result may be published.
- **Verifiability:** No one can falsify the result of the voting.
- **Robustness:** No coalition of voters or authorities (with realistic proportions) can disrupt the election.
- **Receipt-Freeness:** No voter is able to construct a receipt proving the contents of his vote. Verification criteria for this property can be found in [21].
- **Declarability:** It is possible to check if a particular voter has voted (necessary if voting is mandatory).

Requirements Specific for E-Voting

- **Reviseability:** A voter can change their vote. This prevents some forms of coercion, and is only needed if the voting location is not controlled by an election administration.
- **Efficiency:** The election can be administred with a reasonable amount of resources.
- **Convenience:** The voting procedure must be as easy as possible for the voter.

Disputed Requirements

- **Atomic verifiability:** Individual voters can verify that their vote has been properly counted.

  This property is disputed and reconsidered in favour of universal verifiability, since it contradicts the receipt-freeness property.

- **Universal verifiability:** Anyone can verify the tally.

  This property is often seen as too hard to achieve. It is proposed to be weakened, such that a trusted authority can verify the tally, but not every single person. Universal verifiability is hard to implement without violating the privacy and receipt-freeness property (see [17] page 91 and Sect. 3.5 of this document for more information).

3.2 Voting Structure Reference

For reference on which general security should be achieved, we hereby define the traditional voting scenario as it is known in Switzerland. The standard voting-booth paper ballot voting scenario is a four step process:

1. The gouvernement enlist the eligible voters and sends them all information and materials needed for voting.
2. In the voting-booth: The voter authenticates himself to the voting supervision which signs the folded voting intentions as well as a paper stating his voting eligibility.
3. The signed eligibilty paper and the vote is put into different ballots.
4. The votes are counted by a supervised group of random citizen.

Security If the first step, the enlisting, is assumed to be correct; the security critical situation starts when the voter enters the booth. Because of the sign on the back of the voting intention, the vote is made official. Since the paper was folded, the authority does not have a chance to know how the specific citizen was voting, which is necessary to provide *privacy*. To make sure that no one can introduce illicit votes, the proof of eligibility has to be put into another ballot, such that the number of eligible voters can be compared with the number of votes for each voting booth. This guarantees the *eligibility* and *unreusability* as well as *verifiability* by the limit on number of votes, and by knowing who was voting. When tallying the votes, all slips with a mark upon are put aside and will not be counted. This guarantees *receipt-freeness*, since only unmarked votes will be part of the final tally. The final tally is therefore verifiable without giving a possibility for creating a receipt.

3.3 Pricacy within E-Voting Protocols

There are mainly three approaches to achive vote privacy within e-voting: homomorphic encryption, blind signature/anyonmous channel and eligibility tokens.

Homomorphic encryption: Due to the homomorphic property of this encryption function it is possible to compute a tally without decrypting all single ballots, but decrypting all at once ([17] page 26). Homomorphic functions need to have an additive meaning (like a yes/no question) and can not have a set of questions (like a list of candidates).

Compared to the reference structure given in Sect. 3.2 the strength in privacy is equal if the the administrator can not (by any method) decrypt single ballots (since they are accompanied by the identity of the voter, to proof eligibility) and illegal votes can be recognized and put aside.

Blind Signature/Anyonymous Channel: In Sect. 2 we introduced blind signatures. Within e-voting protocols, blind signatures are used to blind the voting intention, while proving the eligibility of the voter. A signing authority can therefore not know what the voter actually votes, but merely sign his voting intention, if the voter is entitled to such. Anonymous channels are used so that the remote voter can not be traced back from the authority that takes the tally, in order to uphold the anonymity granted by the intermediate signing step. See Fig. 2 for a simple illustration on using blind signatures for e-voting.

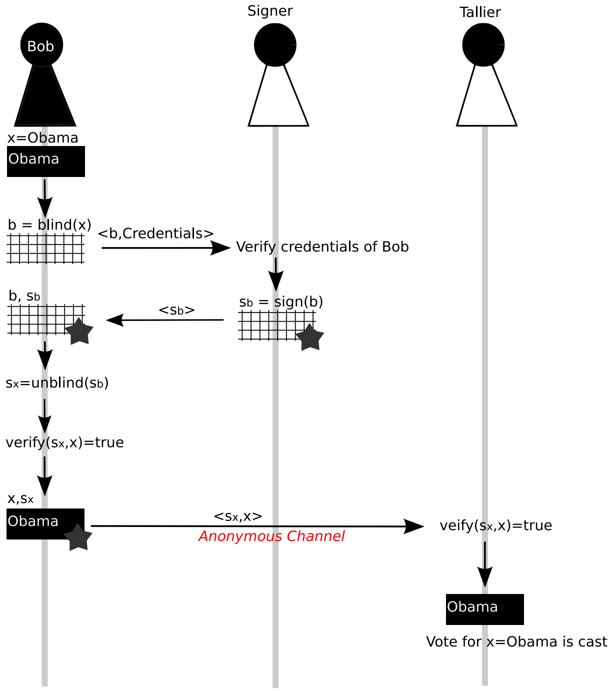

Fig. 2. Simple way of using blind signatures for e-voting. It is too simple to satisfy the requirements needed in real-world voting schemes.

Compared to the reference structure given in Sect. 3.2, the strength in privacy is only equal if either the blind signature scheme used is of the type strong blind signature (see Sect. 2.3) or the validating authority can never see the unblinded message.

Eligibility Token: Schemes based on eligibility token, like the two agency protocol presented by Salomaa et al. in [3] provide privacy by separating the duties of validating and tallying only. The validator hands out eligibility tokens (for example a secret random password) which the tallier will be able to compare against the list of all valid tokens as proof. As long as the tallier and the validator does not collude, none of the authorities can know which person was voting how, which would result in perfect vote secrecy. Nontheless it is a property quite hard to guarantee since both authorities have to speak with each other privately (i.e. exchanging the list of valid tokens). This communication can not be disclosed for verification, since that would break privacy.

3.4 Blind Signatures within E-Voting Protocols

The first time blind signature were proposed for e-voting protocols was 1992 by Fujioka, Okamoto and Ohta in [4]. It is seen as the first practically usable voting scheme. The protocol was published under the name *"A Practical Secret Voting Scheme for Large Scale Elections"* and referenced everywhere with *FOO92* or just *FOO*. It is shown and described in Fig. 3. The protocol is a generic protocol not defining any concrete algorithms (like RSA); only the cryptographic methods to be used are defined.

Their system only aspired to achieve the properties of *completeness*, *soundness*, *privacy*, *unreausability*, *eligibility*, *fairness* and *verifiability*. It managed to do so for most of them, but lacks eligibility in a certain case: If a voter registers, but abstains from the voting phase, the validating agent can add a ballot on the voters behalf. Other properties that were not included, but which are seen today as requirements are missing as well. This includes *receipt-freeness* (see also Sect. 3.5 that discusses this property and its problems) and *convenience* (the voter needs to handle two interacting sessions, and not just one). Modifications to the protocol solved some of the issues later on, but receipt-freeness proved to be hard to achieve. Røsland describes these problematics and proposed solutions in [17] on page 50 in a well-written and summarized manner.

Blind signatures within *FOO* are needed to gain voting eligibility while disclosing the voting intention to the validating agent. The disclosure is only meaningful if the derived signature of the unblinded message does not allow any conclusion on the original blinded signature, resulting in the need of *strong blind signatures* (see Sect. 2.3 for a classification of blind signatures). By having this requirement satisfied, the scheme has perfect vote secrecy. The scheme is specifically strong because even when the validating or the tallying agent disrupt the election by either not counting a ballot or sending a wrong signature, the voter can prove the disruption without revealing his voting intention by showing his unblinded encrypted ballot and the derived signature from the validating agent.

Improvements to FOO There are many papers stating ideas on how to improve *FOO* by different means. As already mentioned [17] gives a good and well-explained overview. The following lists a few ideas on how to solve the

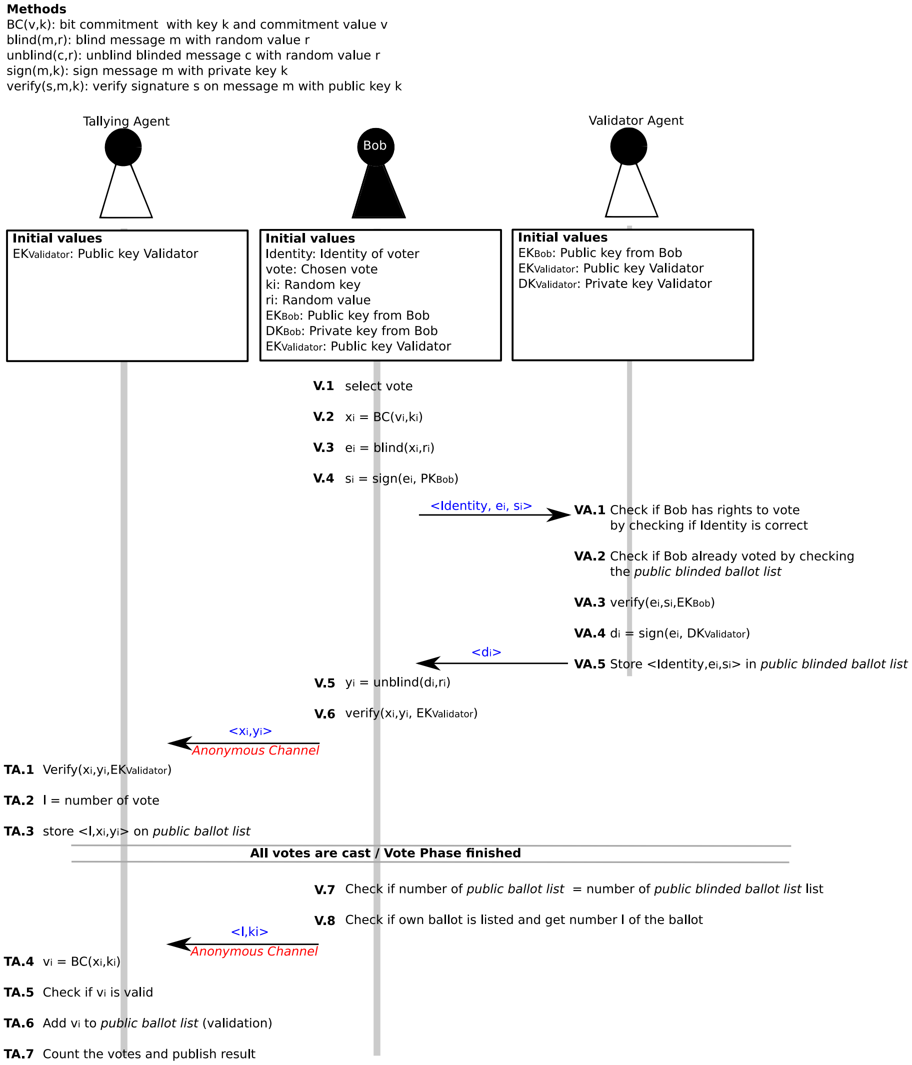
Methods
BC(v,k): bit commitment with key k and commitment value v
blind(m,r): blind message m with random value r
unblind(c,r): unblind blinded message c with random value r
sign(m,k): sign message m with private key k
verify(s,m,k): verify signature s on message m with public key k
Tallying Agent
Bob
Validator Agent
Initial values
$EK_{Validator}$: Public key Validator
Initial values
Identity: Identity of voter
vote: Chosen vote
ki: Random key
ri: Random value
EK_{Bob}: Public key from Bob
DK_{Bob}: Private key from Bob
$EK_{Validator}$: Public key Validator
Initial values
EK_{Bob}: Public key from Bob
$EK_{Validator}$: Public key Validator
$DK_{Validator}$: Private key Validator
V.1 select vote
V.2 $x_i = BC(v_i,k_i)$
V.3 $e_i = blind(x_i,r_i)$
V.4 $s_i = sign(e_i, PK_{Bob})$
$<Identity, e_i, s_i>$
VA.1 Check if Bob has rights to vote by checking if Identity is correct
VA.2 Check if Bob already voted by checking the *public blinded ballot list*
VA.3 $verify(e_i,s_i,EK_{Bob})$
VA.4 $d_i = sign(e_i, DK_{Validator})$
$<d_i>$
VA.5 Store $<Identity,e_i,s_i>$ in *public blinded ballot list*
V.5 $y_i = unblind(d_i,r_i)$
V.6 $verify(x_i,y_i, EK_{Validator})$
$<x_i,y_i>$
Anonymous Channel
TA.1 $Verify(x_i,y_i,EK_{Validator})$
TA.2 l = number of vote
TA.3 store $<l,x_i,y_i>$ on *public ballot list*
All votes are cast / Vote Phase finished
V.7 Check if number of *public ballot list* = number of *public blinded ballot list* list
V.8 Check if own ballot is listed and get number l of the ballot
$<l,k_i>$
Anonymous Channel
TA.4 $v_i = BC(x_i,k_i)$
TA.5 Check if v_i is valid
TA.6 Add v_i to *public ballot list* (validation)
TA.7 Count the votes and publish result

Fig. 3. The voting scheme is divided into six phases (each phase named in italic): During *preparation* (V.1-V.4) the voter fills in a ballot and sends it after blinding and signing to the validator. In the *administration* phase (VA.1-VA.5) the validator signs the message and sends it back to the voter. The voter in the *voting* phase (V.5-V.6) then unblinds the signature and sends it to the tallier that *collects* all votes (TA.1-TA.3). After the vote is finished the voter *opens* his vote by sending his commitment key (V.7-V.8). At last the tallier *counts* the votes and publishes the result (TA.4-TA.7).

particular problems of receipt-freeness, abstaining voter vote injection and convenience. None of these modifications change something on the voter validation with blind signature. There is no change concerning vote secrecy compared to *FOO*.

Solve Abstaining Voter Problem by Secondary Proof: In [7] *"A Practical Electronic Voting Protocol Using Threshold Schemes"* (1994) Baraani-Dastjerdi et al. proposed an improvment to *FOO* by introducing a second eligibility proof (in the form of a sealed pseudonym given to the voter in advance) and a threshold key system on the tallying side. This with the aim to solve the abstaining voter problem. The tallying agent would therefore do a second eligibility validation, which seems to contradict the initial idea of the scheme to separate these two tasks. The scheme is rather complicated and not really usable for real world application.

Achieve Convenience: In [9] *"A Secure and Practical Electronic Voting Scheme for Real World Environments"* (1997) W. Juang and C. Lei tried to solve the convenience problem of the protocol by encrypting the initial ballot not with a secret bit commitment key, but with a public threshold key of a set of candidates. Therefore the voter does not have to send a key after the vote finished but a set of candidates must decrypt the encrypted ballots. Since a threshold key was used at minimum but not more than a defined number of candidates have to take part that the election can be successfully completed (such that no single candidate can disrupt the election). A similar work was done in 1999 by Okhubo et al. (see [13]).

Achieve Convenience and Solve Abstaining Voter Problem: W. Juang et al. proposed in 1998 in *"A Verifiable Multi-Authorities Secret Election Allowing Abstaining from Voting"* [12] an improvement to their original proposal [9]. They achieved assurance that no validator can inject votes from abstaining voters by dividing the trust amongst several validating agents. This is achieved by using a verifiable threshold public-key scheme[1].

Achieve Receipt-Freeness with Zero-Knowledge Proof: In [10] T. Okamoto developed 1996 *"An electronic Voting Scheme"* a scheme based on *FOO* but allowing receipt-freeness and coercion-resistance. This is mainly achieved by using a trapdoor bit commitment method[2] and a non interactive zero-knowledge proof. This allows the tallier to proof that the plaintext ballot published by a voter is linked to a sealed ballot that the tallier agent got through an untappable anonymous channel[3] from the voter. The protocol is neither convenient for the voter nor

[1] See [17] page 33, [12] or http://en.wikipedia.org/wiki/Threshold_cryptosystem for more information on threshold secret sharing schemes.

[2] Trap-door bit commitment scheme allows to open a commitment in different ways, such that an opening in one way can not be a proof of the commitment. See [17] page 41 for more details.

[3] This is needed by the scheme definition to achieve perfect secrecy.

practical for real world implementations since it is not possible to supply every voter with an untappable channel to the tallier.

Achieve Receipt-Freeness with Cyclic List: Zhe Xia and Steve Schneider recently outlined in [22] a scheme titled *"A New Receipt-Free E-Voting Scheme Based on Blind Signature"*, which incorporates receipt-freeness. It needs an untappable transmission channel between voters, signers and talliers. Voters would be presented a list of candidates in a perfectly cyclic list. That list furthermore is presented in (many or all) rotated states. The voter now votes by picking one of the lists and specifying the difference (offset) by which the picked list has to be rotated in order to put the desired candidate at the current top position of the list. Then he also picks a random number, a *trapdoor key* of a trap-door bit commitment scheme. Now the ballot is created and blinded with this information, then signed by the validating agent, unblinded by the voter, and submitted through an anonymous channel to the tallier. The way the voter makes his choice by offset and rotation of the cyclic list, makes it impossible to proof his choice later on, even though the result is verifiable. As the protocol discussed before, it is not practical for real world implementations because of its need for untappable channels.

Implementations of FOO variants There were not only propositions on how to solve specific problems, but also concrete implementations. The most important problem with FOO to be solved for concrete implementations was the convenience problem. By forcing the voters to have two sessions with the authority would lower the acceptance and render it unuseable. Following a description of three interesting and important implementations.

Sensus: In [8] *"Sensus: A Security-Conscious Electronic Poling System for the Internet"* Cranor and Cytron implemented FOO for the Washington University in 1997. The implementation was done using Perl and C on Unix based computers. The used cryptographic library was RSAREF[4]. There are two main differences to *FOO*: To achieve convenience, the voter sends the commitment key to the tallier only after receiving a receipt from him. Furthermore, it needs an additional agent, a registrar, where the voter has to register himself, deposit his public key and in return receive a security token that will be used to identify his identity.

This implementation has severe problems making it unusable for real world use: it does neither solve the abstaining voter problem nor the receipt-freeness. It even explicitly returns a receipt. Furthermore the property of fairness is not given anymore since the tallier can decrypt the ballots before the election ended. It also does not implement anonymous channels therefore privacy is broken (since the used open channels can be tapped and traced). It also makes vote selling very easy since the security token and the public/private keypair can be sold to

[4] RSA reference implementation by RSA labs.

someone. A voter needs not only to trust the client application compiling the voting intention into a ballot but also the tallying agent, the connection and the administrator not to disrupt the election. But also the administration of the election has to trust the votees not to cheat and buy/sell/coerce votes.

EVOX and EVOX-MA The EVOX [11] scheme was originally implemented by Herschberg for the Massachusetts Institute of Technology (MIT) in 1997. Herschberg solved the convenience problem by having an anonymizer service, which collects all the ballots. After the votes are cast the anonymizer mixes them and sends them to the tallier. With this *fairness* is warranted, since no vote can be published before the election ends. The anonymizer is not able to open the ballots, since they are encrypted with the talliers public key. Also privacy is given since tracing the user does not give more information but that the user participated in the election. The only problem still remaining is the that the validating agent can insert votes for an absent voter without being noticed. An interesting point where the scheme differs to *FOO* is that the voter does not create the ballot himself but receives upon request a ballot offering all possible choices given by this election from the validator. This enables the system to run several elections or surveys at the same time.

Based upon EVOX, DuRette created EVOX-MA [14]. There he tried to solve the abstaining voter problem by separating the trust of giving out the ballot and blind signing the ballot to different servers (adding the term of ”managers” for the ballot distributor). Furthermore he defined that the voter does not only blind sign his ballot by one validator, but by several, where the amount of signatures needed must be more than half of the amount of validating servers available, such that it is not possible to reuse the ballot with another set of validators. This is a possible implementation of a threshold cryptosystem, allowing the protocol to be more robust. If one validating agent fails, there are still enough others to go on. It also makes it impossible for validating agents to collect ballots. And because of the many signatures needed (more than half of the existing agents) it is hard to inject votes.

Even though Joaquim et al. state in [15] on page four that injection is still possible when a set of administrators and the manager collude. This ”attack” is based on the fact that the voter gets the same password to contact each of the validating agents. Next to that the system surely also lacks of reciept-freeness. The list of valid ballots are published publicly after the election or survey, and can be used to proof to have voted for a defined cause or person.

REVS REVS [15], developed by Rui Joaquim et al. from the Instituto Superior Tcnico, Lisbon, builds upon EVOX-MA. It has the same general structure, but tries to be more robust when using the scheme for real world elections[5]. The aim is to have a system which satisfies all requirements but receipt-freeness. It is designed such that the voter only has to give trust into the implementation of the client application, but none of the server environments. The changes to

[5] Therefore also the name REVS = Robust Electronic Voting System.

EVOX-MA lie in many small details concerning the implementation and are not explained here. We can recommend their article for its readability and relevance to real world scenarios.

3.5 Verifiability versus Receipt-Freeness

Depending on the kind of voting scheme used, the ballots become atomically verifiable. While this can possibly be a very useful property when it comes to verifying that a tally was correct, right down to the individual level, this can be very dangerous. It may present a voter with a type of "receipt", no different from an actual receipt intended as such. This information is a proof of who voted for what and can be abused to coerce or buy voters consecutively.

Conversely, if one would make the voting entirely receipt-free and unverifiable, coercion wouldn't even be needed anymore - a mistake or corruption in any part of the authority running the vote could entirely falsify the result. This is obviously not a valid option, as it violates all of the principles outlined under Sect. 3.1. Thus, researchers are left with the dilemma of achieving a balance between verifiability and receipt-freeness, two attributes somehow opposed to each other which are, however, both needed for a truly democratic e-voting scheme. This dilemma has yet to be solved in a satisfactory way, especially in the context of real-world elections.

In the following part of this section we will identify three non-exclusive categories of solutions into which research has been invested, exemplarily point out key details and problems of a voting scheme in two of the categories, and give references for further reading.

Avoid Creating any Form of Receipt This is summarily called receipt-freeness. At first glance, this is very simple to achieve by means of simply handing the voter an anonymous eligibility token, and then just letting him/her vote once. Yet, the criteria is much more complex to implement satisfactorily under the assumption that all devices and the transmission channel in between may be compromised, and / or the authority issuing such eligibility tokens can not be trusted. In such a scenario, how could a voter even know that his vote has been recorded correctly and counted, lest that the result of the voting at large is correct?

There are several approaches to avoid creating receipts without loosing verifiability. Building upon blind signatures there are for example [22] by Z. Xia et al. or [10] by T. Okamoto. Both are described in Sect. 3.4 on page 12. Others use homomorphic encryption instead. A few noteworthy examples: *"Efficient Receipt-Free Voting Based on Homomorphic Encryption"* [19]; *"Receipt-Free Homomorphic Elections and Write-in Voter Verified Ballots"* [20].

The problems with the aforementioned schemes are manifold. To take the blind signature examples: They all use untappable channels to guarantee the properties of the scheme. The assumption of a totally safe mix-net for the transmission of data is a problem in practice. It starts with how anyone, especially less the general public - the voters, would be able to audit that it is such without possibly compromising the system in the process, either by knowledge or by manipulation. Also, history is riddled with transmission channels that were supposed to be entirely safe, but were not. Thus the assumption of the channels involved being untappable is not unlikely to be wrong if these were implemented in reality, and with it, the entire scheme would fail to protect any of the properties we mentioned listed as requirements for e-voting.

Furthermore, within these schemes, if the signing and voting authority collaborated, the result would be the same as if the transmission channel had been compromised, with the additional option of forging false votes and discarding real ones. It is worthy of note that this is a generic threat for not-directly voter-verifiable schemes, and especially undetectable at a later point in schemes where no receipts exist.

Avoid Creating Atomic Identifiers Instead of having no receipt, where *verifiability* is very hard to guarantee, one can try to just avoid atomic identifiers. To achieve this, signatures can be constructed in a way that only audits in larger batches are possible, or that otherwise no link can be established between a ballot and voter. These two methods are not exclusive and even frequently practiced together. The former, making audits possible only in batches, is done by a mix-net[6] with ballots that do not contain anything that directly identifies the voter. The trace to the voter is thus lost ”inside” the mix-net. The problem here is that voter eligibility is not ensured in the process; this is generally in the domain of the next described method.

The latter, erasing the link between ballot and voter, is widely practiced in many nation's real world paper ballot elections, where a voter first shows a sort of document to prove eligibility, and then is granted a ballot to cast (near-) anonymously. How it works is explained in our reference voting system in Sect. 3.2. Even so, while in real elections it is not (yet?) deemed a great risk that the voter generally leaves fingerprints, his DNA, and handwriting on both the ballot and the proof of eligibility, such marks are unfortunately much more dangerous in their electronic equivalents, as they generally are much easier to analyze. This is addressed to a large extent in this report concerning blind signatures schemes, and left undiscussed for homomorphic encryption schemes[7].

Make Receipts Hard to Abuse There are also attempts to make receipts hard to abuse in current real-world scenarios while providing sufficient methods to give receipts and perform verification. Being a very broad category, this would not offer any absolute or near-absolute guarantees, but is closest to the status quo on paper ballot voting. Arguably, it could also be more easily adapted to comply with additional requirements such as regional needs or voting laws.

[6] See description in Sect. 1.2 or [17], page 20ff for more information.

[7] A good introduction to homomorphic schemes for e-voting can be found in [17], page 26ff.

Chaum, for instance, devised in [16] a *voter-verifiable* receipt scheme that hides the receipt information in multiple pieces. The idea is that receipt type of information is rasterized, encrypted and printed on several pieces of paper. To secure the printouts they get laminated. First the voter would confirm validity of the stack of laminated papers by means of visual comparison for equality. Then, some parts of the stack will be held onto by the voter, others will go to the voting authorities. Without all the parts combined, no information could be gained.

Yet, while they seem realistically useful in making coercion or vote manipulation more difficult than before, the aforementioned approach does not allow for blinding, and has related severe drawbacks. The same authority would both be confirming identity and taking the tally at the same time, with no guarantee of anonymity - the authority would have to be *trusted*.

As a further consequence of the whole, decipherable information being available during receipt generation and verification, it is also susceptible to being captured by *video surveillance* and compromised signature generating machines put in place by other parties than the voting authorities.

4 Outlook

While it can be observed that the continued academic debate and intense studies have not yet led to widespread adoption of e-voting systems, the interest in e-voting is unabated; and no significant dispute on the potential of e-voting has been voiced. However, requirements in different countries and regions vary, as indeed the very forms of democratic governments do. We believe that, while the core of many recent e-voting schemes is fairly solid by now, the proper adaptation of these to regional needs would still be a very non-trivial task. Given the bad reputation hit involved in botching a vote, it is easy to see why most administrations take a conservative approach to actually implementing e-voting, especially after the public has already developed a sensitivity to the issue in the wake of the various e-voting related irregularities reported in the United States. It thus is still fairly open whether we will see more wide-spread adoption of e-voting within two, five, ten or twenty years, but it is very likely that it will happen.

References

1. D. Chaum: Blind Signatures for Untraceable Payments. Proceedings of CRYPTO '82, pages 199-203 (1982)
2. David Chaum: Blind Signature Systems, Advances in Cryptology. Crypto '83, Plenum (1983)
3. H. Nurmi, A. Salomaa, and L. Santean: Secret Ballot Elections in Computer Networks. Computers & Security, 36(10) pages 553-560 (1991)
4. Atsushi Fujioka and Tatsuaki Okamoto and Kazuo Ohta: A Practical Secret Voting Scheme for Large Scale Elections. ASIACRYPT '92: Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, pages 244-251 (1993)

5. M. Franklin and M. Yung: Blind Weak Signatures. Advances in Cryptology, Proceedings EUROCRYPT '94 pages 67-76 (1994)
6. Patrick Horster and Holger Petersen: Classification of Blind Signaure Schemes and Examples of Hidden and Weak Blind Signatures. Technical Report, University of Technology Chemnitz-Zwickau (1994)
7. A. Baraani-Dastjerdi, J. Pierprzyk and R. Safavi-Naini: A Practical Electronic Voting Protocol Using Threshold schemes. Technical Report, University of Wollongong, Australia (1994)
8. L. Cranor and R. Cytron: Sensus: A Security-Conscious Electronic Polling System for the Internet. Master's thesis, Washington University (1997)
9. W. Juang and C. Lei: A Secure and Practical Electronic Voting Scheme for Real World Environments. IEICE Trans on Fundamentals, E80-A(1), pages 64-71 (1997)
10. Tatsuaki Okamoto: Receipt-Free electronic Voting Schemes for Large Scale Elections. Proceedings of Security Protocols Workshop, pages 25-37 (1997)
11. M. A. Herschberg: Secure Electronic Voting Over the World Wide Web. Master's thesis, Massachusetts Institute of Technology, USA (1997)
12. W. Juang, C. Lei and P. Yu: A Verifiable Multi-Authorities Secret Election Allowing Abstaining from Voting. The Computer Journal, Vol. 45, No. 6, pages 672-682 (1998)
13. M. Ohkubo, F. Miura, M. Abe, A Fujioka and T. Okamoto: An Improvement on a Practical Secret Voting Scheme. Information Security'99 LNCS Vol. 1729, pages 225-234 (1999)
14. B. W. DuRette: Multiple Administrators for Electronic Voting. Bachelor thesis, Massachusetts Institute of Technology, USA (1999)
15. R. Joaquim, A. Zuquete and P. Ferreira: REVS - A Robust Electronic Voting System. IADIS Internations Conference e-Society, pages 95-103 (2003)
16. D. Chaum: Secret-Ballot Receipts: True Voter-Veriable Elections. IEEE Security & Pricacy 1540-7993/04 - www.computer.org/security/ (2004)
17. Geir Røsland: Remote Electronic Voting. Hovedoppgave, University of Bergen, Norway (2004)
18. C. Nielson, E. Andersen and H Nielson: Static Validation of a Voting Protocol. Electronic Notes in Theoretical Computer Science 135-1, pages 115-134 (2005)
19. Martin Hirt, Kazue Sako: Efficient Receipt-Free Voting Based on Homomorphic Encryption. Advances in Cryptology EUROCRYPT 2000, pages 539-556 (2000)
20. Alessandro Acquisti: Receipt-Free Homomorphic Elections and Write-in Voter Verified Ballots. CMU-ISRI-04-116 (2004)
21. Delaune, S. Kremer , S. Ryan, M. Lab: Coercion-Resistance and Receipt-Freeness in Electronic Voting. Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW'06), IEEE Comp. Soc. Press (2006)
22. Zhe Xia, Steve Schneider: A New Receipt-Free E-Voting Scheme Based on Blind Signature (Abstract). Proceedings of Workshop on Trustworthy Elections (WOTE 2006), pages 127-135 (2006)
23. What is a blind signature scheme? http://www.rsa.com/rsalabs/node.asp?id=2339