SECURITY & PRIVACY COLUMN

MATTEO MAFFEI, CISPA, Saarland University
maffei@cs.uni-saarland.de

The technical column on Security and Privacy of the SIGLOG newsletter kicks off with two invited contributions.

— Cătălin Hriţcu (INRIA, Paris-Rocquencourt) reports on the recently organized "Joint EasyCrypt-F*-CryptoVerif School 2014". These three state-of-the-art security verification tools constitute an example of how logic and program verification may bring crucial contributions in the security domain. The presented material is available online and is certainly of great interest for the readers of the SIGLOG newsletter.

— Véronique Cortier (Loria) overviews a topic that is of public interest and constitutes an extraordinary research opportunity for logic and program verification researchers, namely the security of electronic voting. Véronique's work demonstrates the effectiveness of formal methods in rigorously reasoning about the security of cryptographic protocols in general, and electronic voting systems in particular. In this column, Véronique reviews the state of the art in this field, pointing out research challenges of particular interest for the SIGLOG community.

A special thanks to both authors for accepting the invitation and for their great contributions!

Formal verification of E-voting: solutions and challenges

Véronique Cortier, CNRS - Loria, France

©INRIA/Photo Kaksonen

In the last ten years, electronic voting has been used in an ever growing number of elections. There are many reasons for this development. First, some election modes require a mechanized way for counting since the number of questions or the number of choices is too large for a manual counting. Electronic voting also allows one to vote from home, possibly avoiding long travels. Sometimes, it simply follows the trend of using Internet in our daily life.

Of course, the use of electronic voting raises many questions: How to make sure my vote will be counted? Does my vote remain private? How to trust the final result? Is it possible to buy votes on the Internet? While some countries such as Estonia, France, or Norway use electronic voting for legally binding elections, some other countries have banned e-voting at least for some time. This is the case for example of the United Kingdom, Netherlands, or Germany [Barrat et al. 2012]. Electronic voting is used nevertheless and is likely to be used in most developed countries worldwide in non legally binding elections like the election of many representative committees (administration councils, scientific councils, etc.). Some of these elections may actually involve a large number of voters (more than a million) and may have important issues for example when it comes to elect the leader of a political party.

There is therefore an urgent need to offer secure, reliable, and trustworthy e-voting systems. The research community in logics and program verification has already a long tradition in developing analysis techniques for security protocols. E-voting protocols raise however new research challenges.

1. PROPERTIES

What is a good e-voting system? This a wide question and we focus here only on security properties the design of the system should offer. We will not discuss implementation or usability issues.

1.1. Privacy properties

A first family of properties is related to *privacy*. Any voting system should ensure that how voters voted is not leaked to anyone. To this end, most voting systems encrypt the votes on the voter's computer with the public key of the election, before sending it to the server. Note however that it is very difficult to protect against compromised computers that may leak the choice of the voter to a malicious third party. One possible countermeasure is to use pre-received code sheets [Chaum 2001] to hide the actual choice from the computer. Other currently studied solutions make use of (not necessarily trusted) external secure devices that perform parts of the computation [Lipmaa 2014].

Protecting vote privacy is necessary but not always sufficient from a security point of view. Ideally, a voter should not be able to show how he voted even if he is willing to. This is to avoid vote buying or coercion: if a voter can prove how he voted, then he may

be coerced to vote in a certain way or he may sell his vote. Systems resistant to such attacks are called *receipt-free* or *coercion resistant*.

Another dimension of privacy is *everlasting privacy*. Will my vote remain secret in twenty years? It could be very embarrassing if our votes could be revealed years later. Most voting systems rely on cryptography which itself relies on computationally difficult problems. It is likely that in twenty years the keys in use nowadays will be broken, either due to faster computers, or algorithms improvements, or the discovery of some implementation flaws, and possibly a combination of those. It is therefore important for e-voting systems to also be robust on the long term, even when keys are eventually lost.

1.2. Verifiability properties

Anyone should be able to check that the final announced result does count all the votes casted by the voters. This *end-to-end verifiability* property may be divided in several sub-properties.

Individual verifiability ensures that each voter can check that his ballot appears on the ballot box. A natural way to achieve individual verifiability is to allow the ballot box to be publicly available on a website. But even if the ballot box is not available to the voters, some public version of the ballot box may suffice for a voter to check that his ballot has been successfully carried to the box. An example of such a case is a voting system based on Pedersen commitments [Cuvelier et al. 2013], that aims at everlasting privacy.

Then the announced result should correspond to the ballots on the box. This property is often referred to as *universal verifiability*. Typically, talliers produce a (zero-knowledge) proof of correct decryption. To preserve anonymity while offering verifiability, the tally is either computed after shuffling the ballots through mixnets or using homorphic encryption (e.g. El Gamal), which allows anyone to combine the votes before decryption.

Lastly, it should be possible to check that only legitimate voters have voted, otherwise the voting system may be subject to ballot stuffing. This is called *eligibility verifiability*.

Verifiability ensures that the entire process can be watched. Not only it is a very desirable property, but it also alleviates the task of auditing the implementation. There is no need anymore to trust the entire code run on the election system. The idea is that if something goes wrong it will be eventually noticed. It is even better if the faulty behaviour can be identified. If something went wrong, who should be blamed? This notion of *accountability* has been developed for example in [Küsters et al. 2012] and is very useful in the context of voting protocols.

2. EXISTING E-VOTING SYSTEMS

We briefly discuss some of the main voting systems in use and we present one of them in more details, Helios.

2.1. Brief survey on existing e-voting systems

The *Estonian voting system* is a rather simple system: voters encrypt their votes and sign the corresponding ballot, relying on the fact that Estonian citizens have an electronic ID card that contains a signing key. A voter may later query back his ballot to check that it is part of the box. But there is no verifiability: voters cannot check that their vote is actually counted nor that the result corresponds to the bulletin board. A precise description of the system and its vulnerabilities can be found in [Springall et al. 2014].

The *protocol used in Norway* is developed by Scytl [Gjøsteen 2010; doc 2013]. Once they have voted, voters receive a code by SMS, which can be used to check that their ballot belong to the box. Voters may still not verify the announced result. However, this protocol offers some "proxy-verifiability": external (approved) auditors can check the consistency of the ballot box and can check the proofs produced by the tallier when decrypting the ballots.

Helios [Adida et al. 2009] is an academic protocol developed by Ben Adida [Adida 2008], based on a protocol proposed by Cramers *et al* [Cramer et al. 1997]. It is used to elect the IACR board (International Association for Cryptologic Research) since 2010 [IACR]. The ballot box is public so that anyone can check that his ballot is present. The final result is also verifiable, based on either ElGamal encryption or mixnet tallying (both options are available). Helios therefore offers both ballot privacy, individual, and universal verifiability. A variant of Helios, Belenios [Cortier et al. 2013], further guarantees eligibility verifiability. Note however that neither Helios nor Belenios are receipt-free: these systems should be used in low-coercion environments.

Civitas [Clarkson et al. 2008] is one of the only protocols (if not the only one) to offer both verifiability and coercion-resistance. When a voter is under coercion, he may vote as prescribed using a fake credential. Once the coercer has left, the voter may vote again, with a valid credential. A coercer may not distinguish a fake credential from a valid one. Then invalid ballots are discarded after some shuffling, to guarantee that a coercer can still not notice any difference. Anyone can check that only invalid ballots have been rejected. However, to our knowledge, this protocol has not been used in real elections due to some practical issues. In particular, getting rid of invalid ballots requires a computation whose complexity is quadratic in the number of submitted ballots.

We have focused here on purely electronic voting systems, where voters cast their votes using the Internet. There are also a variety of hybrid systems where voters vote at a polling station and are offered some means to check that their votes have been counted. Such systems include for example Scantegrity [Chaum et al. 2008] and Prêt-à-voter [Ryan et al. 2009].

2.2. The Helios protocol in more details

An e-voting system should guarantee both privacy (no one know how I voted) and verifiability (I can check that my vote has been counted). These two properties are conflicting and the design of an e-voting system requires a good balance between confidentiality and verifiability. We explain how these two properties can be matched through the example of the Helios protocol [Adida et al. 2009]. In Helios, the ballot box is public (typically a web page) and anyone can access to it. Let pk denotes the public key of the election. The corresponding decryption key sk is split among several authorities: all the authorities need to collaborate to decrypt a message. For the sake of robustness, it is actually better to use *threshold decryption*: only k out of n authorities are needed to decrypt. This avoids to cancel a whole election because some authority has lost its key.

For simplicity, we assume here a referendum election where voters vote for 0 or 1. Assume Alice wishes to vote v_A. She simply encrypts her vote with the public key of the election, yielding the message $\{v_A\}_{pk}$. She also appends a zero-knowledge proof ZKP_A that v_A is a valid vote, that is $v_A = 0$ or $v_A = 1$. This is to avoid that Alice picks a wrong voting option. We will see later why it is important. Then Alice simply sends her ballot $\{v_A\}_{pk}, ZKP_A$ to the ballot box. Since the ballot box is public, she can check that her ballot is present on the ballot box, among other ballots as illustrated above.

	Ballot Box
Alice	$\{v_A\}_{pk}, ZKP_A$
Bob	$\{v_B\}_{pk}, ZKP_B$
Charlie	$\{v_C\}_{pk}, ZKP_C$

The encryption scheme used in Helios is El Gamal. The tally phase makes use of the homomorphic property of El Gamal: The multiplication of the encryption of the votes yields the encryption of the sum of the votes.

$$\prod_{i=1}^{n} \{v_i\}_{pk} = \{\sum_{i=1}^{n} v_i\}_{pk}$$

We can observe here why voters must prove that they vote either 0 or 1. If they could submit an arbitrary message, they could modify arbitrarily the result by sending the encryption of k where k is a positive integer if they wish to increase the score and k is a negative integer if they wish to decrease the score.

Once $\{\sum_{i=1}^{n} v_i\}_{pk}$ is computed (which can be performed by anyone), the decryption authorities simply need to decrypt this single message, which preserves voter's privacy: the link between ballots and votes is never leaked. The authorities also provide a (zero-knowledge) proof of correct tabulation. Helios is therefore verifiable: any voter can check that his ballot belongs to the ballot box and everyone can check that the result has been correctly computed.

Helios is actually not fully private as it is subject to ballot-copying [Cortier and Smyth 2011]. A voter can copy a ballot on the board and send it as his own ballot. For simplicity, let's consider an election with three voters, Alice, Bob, and Charlie where Charlie is a dishonest voter. Assume that Alice and Bob have already voted.

	Ballot Box
Alice	$\{v_A\}_{pk}, ZKP_A$
Bob	$\{v_B\}_{pk}, ZKP_B$

Then Charlie can simply copy Alice's ballot $\{v_A\}_{pk}, ZKP_A$ and send it to the ballot box, pretending it is his ballot.

	Ballot Box
Alice	$\{v_A\}_{pk}, ZKP_A$
Bob	$\{v_B\}_{pk}, ZKP_B$
Charlie	$\{v_A\}_{pk}, ZKP_A$

The tally reveals the result of the election $r = 2v_A + v_B$. Charlie can then easily deduce Alice's vote: if $r \leq 1$ then $v_A = 0$ otherwise $v_A = 1$.

This attack can be fixed by weeding or rejected duplicated ballots. [Bernhard et al. 2012] rigorously shows that this is indeed sufficient.

3. SECURITY ANALYSIS

As for more traditional security protocols, the design of e-voting protocols is difficult and error-prone. These systems need a precise modelling and a rigorous analysis. The first task consists in formally defining security properties.

3.1. Properties

For vote privacy, we need to express that an adversary does not learn how a voter voted. However, a voting system does leak some information since the final result depends on the votes. For example, in the extreme case of unanimity, there is no privacy left. One common way of defining vote privacy is to require that an adversary cannot distinguish

from the scenario where Alice is voting v_1 and Bob is voting v_2 from the converse scenario where Alice is voting v_2 and Bob is voting v_1:

$$\mathsf{Voter}(A, v_1) \mid \mathsf{Voter}(B, v_2) \approx \mathsf{Voter}(A, v_2) \mid \mathsf{Voter}(B, v_1)$$

This property has been formalized in symbolic models [Delaune et al. 2009], where primitives are abstracted by terms, as well as in cryptographic models. Cryptographic models offer three main types of definitions

— **Game-based definitions.** Game-based definitions are usually convenient for proving security since they are well tailored to proof by reduction to security assumptions (such as integer factorisation). The cryptographic counter-part of the privacy definition stated above has been proposed in [Benaloh and Yung 1986]. This definition does not capture all the information an attacker can get. For example, in the case of an approval voting, it may be the case that an attacker may distinguish when Alice is voting 0 and Bob is voting 2 from the scenario where both Alice and Bob are voting 1. Therefore several more general game-based definitions of privacy have been recently proposed (eg [Küsters et al. 2010; Bernhard et al. 2011; Bernhard et al. 2012; Chase et al. 2013; Cuvelier et al. 2013; Bernhard and Smyth 2013]). Surprisingly, game-based definitions are difficult to get right, which explains the number of definitions.
— **Entropy-based definitions.** Another approach intends to capture all kind of information leaked by a voting protocol. This is the case of entropy-based privacy definitions [Coney et al. 2005; Bernhard et al. 2012]. These definitions capture several sources of privacy leakage: leakage from the protocol itself but also from the votes distribution, or the result function: telling only the name of the elected candidate is clearly less leaky than providing the entire set of votes. Entropy-based notions are therefore powerful but also much harder to prove.
— **Ideal functionality.** Finally, privacy can be defined through an ideal voting system (also called ideal functionality) and proving that the actual voting system does not provide anymore information than its ideal version [Groth 2004].

As explained earlier, vote privacy is actually a weak form of privacy. For elections with important issues, voting schemes should be receipt-free or even coercion-resistant: an adversary should not be able to detect when a voter under coercion still votes for the candidate of his choice instead of voting as indicated by the coercer. In symbolic models, receipt-freeness as well as coercion-resistance can also be expressed through equivalence properties [Delaune et al. 2009; Backes et al. 2008]. Intuitively, these definitions state that an adversary should not be able to distinguish between the case where the voter votes exactly as requested by the coercer from the case where the voter uses a strategy to vote as he wishes, possibly revoting when he is not under coercion anymore.

Surprisingly, verifiability has deserved less attention than privacy. Individual, universal, and eligibility verifiability have been defined in symbolic models [Kremer et al. 2010]. This work also shows how these definitions depend on each other. Cryptographic models usually define verifiability in a more global way such as in [Juels et al. 2010; Cortier et al. 2013] where it is simply stated that the result should include at least the votes of all honest voters.

3.2. Formal analysis

Once the desired properties are defined, a first approach is to prove security by hand. This has been done for the Helios protocol for example, both in symbolic models [Cortier and Smyth 2013] and in cryptographic ones [Bernhard et al. 2011; Bernhard et al. 2012]. However, these proofs are rather long and difficult to check. We therefore focus here on how to automate, at least partially, security proofs of e-voting

protocols. Proofs in computational models are difficult to automate. In contrast, several mature tools exist for the symbolic analysis of security protocols. For example, ProVerif [Blanchet 2005; Blanchet et al. 2005], Scyther [Cremers 2008], as well as Avispa [Armando et al. 2005] have been successfully applied to many protocols of the literature, including widely deployed ones such as TLS [Bhargavan et al. 2008] or Single-Sign-On [Armando et al. 2008]. However, these tools do not apply well to e-voting systems.

A first reason lies in the properties that need to be analysed. Standard security properties such as authentication or confidentiality are expressed as trace properties: for any execution trace of the protocol, the secret data a remains inaccessible (confidentiality); or for any execution trace, whenever Alice finishes a session supposedly with Bob then Bob has indeed initiated a session with Alice (authentication). Many variants can be defined as trace properties. In contrast, privacy properties are stated as indistinguishability properties, formally expressed through behavioural equivalence properties (e.g. observational equivalence, trace equivalence, or may-testing equivalence [Abadi and Gordon 1997]). Only the ProVerif tool can handle such equivalence properties. It actually proves a stronger notion, called diff-equivalence [Blanchet et al. 2005], that is unfortunately too strong when applied to voting protocols. The ProSwapper tool [Smyth] may help ProVerif in some cases. Note that equivalence properties are not only useful in the context of e-voting, they are also used for other privacy properties such as untraceability [Brusó et al. 2010] or anonymity [Cheval et al. 2013; Arapinis et al. 2014]. Therefore, some prototypes tools dedicated to equivalence have been recently developed: AKISS [Chadha et al. 2012], APTE [Cheval 2014], and SPEC [Dawson and Tiu 2010].

However, all these tools and techniques face a second challenge: e-voting systems make a wide use of not so standard cryptographic primitives. For example, the FOO protocol [Fujioka et al. 1992] relies on blind signatures: voters have their blinded ballots signed by the registration authority and then remove the blinding factor. This property can be formalised as follows [Delaune et al. 2009]:

$$\mathsf{unblind}(\mathsf{sign}(\mathsf{blind}(x,z),y),z) = \mathsf{sign}(x,y)$$

This equation is already out of reach of the APTE and SPEC but this protocol can actually by analysed by Akiss [Chadha et al. 2012]. Consider now the Helios protocol presented earlier: combining encrypted votes yields the encryption of the sum of the votes. This can be reflected by the following equation

$$\mathsf{enc}(pk;v_1) * \mathsf{enc}(pk;v_2) = \mathsf{enc}(pk;v_1+v_2)$$

where $*$ and $+$ are associative and commutative (AC) operators. AC operators are currently out of reach of any existing tool for analysing security protocols. Things get even worse with the Norwegian protocol that uses an ad-hoc construction. In this protocol, voters encrypt their votes with public key $\mathsf{pk}(a_1)$ using ElGamal encryption. The ballot box then re-encrypts a blinded version of the ballot with a secret key a_2 such that the receipt generator can now decrypt the ballot with $a_3 = a_1 + a_2$, without having access to the vote. This is reflected by the following symbolic equations [Cortier and Wiedling 2012]:

$$\mathsf{renc}(\mathsf{enc}(x_{plain},x_{rand},\mathsf{pk}(x_{sk})),y_{sk}) = \mathsf{enc}(x_{plain},x_{rand},\mathsf{pk}(x_{sk}+y_{sk}))$$

$$\mathsf{dec}(\mathsf{blind}(\mathsf{enc}(x_{plain},x_{rand},\mathsf{pk}(x_{sk})),x_{blind}),x_{sk}) = \mathsf{blind}(x_{plain},x_{blind})$$

Coming up with resolution procedures that handle such complex equational theories is a challenging research goal. In particular, we need new techniques for handling AC operators, larger theories, and for combining theories.

Challenges

In the previous section, we have already discussed two main challenges: to evaluate e-voting systems, formal analysis techniques have to cope with both equivalence properties and complex equational theories. No automatic prover can do both for the moment.

It will be probably difficult if not impossible to get decidability results for both equivalence properties and wide classes of equivalence properties. One way to circumvent decidability issues is to design good over-approximations. Defining over-approximations is somewhat easy in the case of trace-based properties: if a protocol P is "simplified" into P' then the security of P' implies the security of P as soon as the execution traces of P' include all execution traces of P. Typical approximations consist in removing nonce freshness (the same nonce may be generated twice) or providing more knowledge to the attacker. However, this reasoning no longer works in the case of equivalence properties. If P is simplified into P' and Q into Q', the equivalence $P' \approx Q'$ does not imply $P \approx Q$. It is therefore necessary to design new approximations, sound w.r.t. equivalence properties. One approach could be to use type systems [Barthe et al. 2014], which proved successful to handle indistinguishability properties as well as a relatively large class of cryptographic schemes.

Surprisingly, privacy has deserved much more attention than verifiability, both in terms of definitions and analysis, while both properties are equally desirable and important for e-voting systems. One explanation is that verifiability seems easier to define. Yet, it remains to formally define verifiability and check the definition against most existing protocols. From a verification point of view, a second step is to study how to automate verifiability analysis. Verifiability can be classified as a trace-based property, for which many techniques exist. However, verifiability requires to count: "the result contains exactly once each vote of honest voters, plus at most k dishonest votes", where k is the number of corrupted voters, that is the number of voters under the control of the attacker. It is unclear how existing tools can cope with verifiability.

Finally, e-voting systems also question the security models that are usually considered for protocols. In particular, it is not always realistic to trust the computer's voters since it may host worms for example. To control the computer voter's behaviour, some e-voting systems make use of external devices (e.g. secure tokens as in banking applications), code sheets, or out-of-band channels (e.g. SMS). Of course, the cost of each of these techniques differ and corresponding precise security models still need to be defined. The threat scenario also differs depending on the election. A challenging question is how to define a voting systems, or a family of voting systems, that can (provably) cope with each threat scenario.

Acknowledgments. I would like to thank Matteo Maffei for his careful reading and his numerous and helpful suggestions.

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no 258865, project ProSecure.

REFERENCES

M. Abadi and A. D. Gordon. 1997. A Calculus for Cryptographic Protocols: The Spi Calculus. In *Proc. of the 4th ACM Conference on Computer and Communications Security (CCS'97)*. ACM Press, 36–47.

Ben Adida. 2008. Helios: Web-based Open-Audit Voting. In *USENIX Security'08: 17th USENIX Security Symposium*. USENIX Association, 335–348.

Ben Adida, Olivier de Marneffe, Oliver Pereira, and Jean-Jacques Quisquater. 2009. Electing a university president using open-audit voting: Analysis of real-world use of Helios. In *Proceedings of the 2009 conference on Electronic voting technology/workshop on trustworthy elections*.

M. Arapinis, L. Mancini, E. Ritter, and M. Ryan. 2014. Privacy through pseudonymity in mobile telephony systems. In *21st Annual Network and Distributed System Security Symposium (NDSS'14)*.

A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P.-C. Héam, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron. 2005. The AVISPA Tool for the automated validation of internet security protocols and applications. In *17th International Conference on Computer Aided Verification, CAV'2005 (Lecture Notes in Computer Science)*, Vol. 3576. Springer, 281–285.

Alessandro Armando, Roberto Carbone, Luca Compagna, Jorge Cuellar, and Llanos Tobarra Abad. 2008. Formal Analysis of SAML 2.0 Web Browser Single Sign-On: Breaking the SAML-based Single Sign-On for Google Apps. In *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering (FMSE 2008)*. 1–10.

Michael Backes, Catalin Hritcu, and Matteo Maffei. 2008. Automated Verification of Electronic Voting Protocols in the Applied Pi-calculus. In *Proceedings of 21st IEEE Symposium on Computer Security Foundations (CSF 2008),*. IEEE, 195–209.

Jordi Barrat, Esteve, Ben Goldsmith, and John Turner. 2012. *International Experience with E-Voting*. Technical Report. Norwegian E-Vote Project.

Gilles Barthe, Cédric Fournet, Benjamin Grégoire, Pierre-Yves Strub, Nikhil Swamy, and Santiago Zanella Béguelin. 2014. Probabilistic Relational Verification for Cryptographic Implementations. In *41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'14)*. 193–206.

Josh Cohen Benaloh and Moti Yung. 1986. Distributing the Power of a Government to Enhance the Privacy of Voters (Extended Abstract). In *Proc. 5th Annual ACM Symposium on Principles of Distributed Computing (PODC'86)*. ACM, 52–62.

David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. 2011. Adapting Helios for provable ballot secrecy. In *Proceedings of the 16th European Symposium on Research in Computer Security (ESORICS'11) (Lecture Notes in Computer Science)*, Vol. 6879.

David Bernhard, Véronique Cortier, Olivier Pereira, and Bogdan Warinschi. 2012. Measuring Vote Privacy, Revisited. In *19th ACM Conference on Computer and Communications Security (CCS'12)*. ACM.

David Bernhard, Olivier Pereira, and Bogdan Warinschi. 2012. How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. In *Advances in Cryptology - (ASIACRYPT 2012)*. 626–643.

David Bernhard and Ben Smyth. 2013. Ballot privacy and ballot independence coincide. In *Proceedings of the 18th European Symposium on Research in Computer Security (ESORICS'13) (Lecture Notes in Computer Science)*, Springer (Ed.).

Karthikeyan Bhargavan, Ricardo Corin, Cédric Fournet, and Eugen Zalinescu. 2008. Cryptographically Verified Implementations for TLS. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*. 459–468.

Bruno Blanchet. 2005. An Automatic Security Protocol Verifier based on Resolution Theorem Proving (invited tutorial). In *20th International Conference on Automated Deduction (CADE-20)*.

Bruno Blanchet, Martín Abadi, and Cédric Fournet. 2005. Automated Verification of Selected Equivalences for Security Protocols. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005)*. IEEE Computer Society, 331–340.

Mayla Brusó, Konstantinos Chatzikokolakis, and Jerry den Hartog. 2010. Formal verification of privacy for RFID systems. In *CSF'10: 23rd Computer Security Foundations Symposium*. IEEE Computer Society, 75–88.

Rohit Chadha, Ştefan Ciobâcă, and Steve Kremer. 2012. Automated verification of equivalence properties of cryptographic protocols. In *21th European Symposium on Programming (ESOP'12) (LNCS)*. Springer. To appear.

Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. 2013. Verifiable Elections That Scale for Free. In *16th International Conference on Practice and Theory in Public-Key Cryptography (PKC 2013) (Lecture Notes in Computer Science)*, Vol. 7778. Springer, 479–496.

David Chaum. 2001. Sure vote: Technical overview. In *Proceedings of the Workshop on Trustworthy Elections (WOTE 01)*.

David Chaum, Aleks Essex, Richard Carback, Jeremy Clark, Stefan Popoveniuc, Alan Sherman, and Poorvi Vora. 2008. Scantegrity: End-to-End Voter-Verifiable Optical-Scan Voting. *IEEE Security and Privacy* 6, 3 (2008), 40–46.

Vincent Cheval. 2014. APTE: an Algorithm for Proving Trace Equivalence. In *Proceedings of the 20th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'14) (Lecture Notes in Computer Science)*. Springer.

Vincent Cheval, Véronique Cortier, and Antoine Plet. 2013. Lengths may break privacy – or how to check for equivalences with length. In *Proceedings of the 25th International Conference on Computer Aided Verification (CAV'13) (Lecture Notes in Computer Science)*, Vol. 8043. Springer, 708–723.

Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. 2008. Civitas: Toward a Secure Voting System. In *Proc. IEEE Symposium on Security and Privacy*. 354–368.

Lillie Coney, Joseph L. Hall, Poorvi L. Vora, and David Wagner. 2005. Towards a Privacy Measurement Criterion for Voting Systems. In *In National Conference on Digital Government Research*.

Véronique Cortier, David Galindo, Stephane Glondu, and Malika Izabachene. 2013. A generic construction for voting correctness at minimum cost - Application to Helios. Cryptology ePrint Archive, Report 2013/177. (2013).

Véronique Cortier and Ben Smyth. 2011. Attacking and fixing Helios: An analysis of ballot secrecy. In *24th Computer Security Foundations Symposium (CSF'11)*. IEEE Computer Society.

Véronique Cortier and Ben Smyth. 2013. Attacking and fixing Helios: An analysis of ballot secrecy. *Journal of Computer Security* 21, 1 (2013), 89–148. DOI:http://dx.doi.org/10.3233/JCS-2012-0458

Véronique Cortier and Cyrille Wiedling. 2012. A formal analysis of the Norwegian E-voting protocol. In *Proceedings of the 1st International Conference on Principles of Security and Trust (POST'12) (Lecture Notes in Computer Science)*, Vol. 7215. Springer, 109–128. DOI:http://dx.doi.org/10.1007/978-3-642-28641-4_7

Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. 1997. A Secure and Optimally Efficient Multi-Authority Election Scheme. In *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'97)*. 103–118.

Cas Cremers. 2008. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, USA, Proc. (Lecture Notes in Computer Science)*, Vol. 5123/2008. Springer, 414–418. DOI:http://dx.doi.org/10.1007/978-3-540-70545-1_38

Edouard Cuvelier, Olivier Pereira, and Thomas Peters. 2013. Election Verifiability or Ballot Privacy: Do We Need to Choose?. In *Proceedings of the 18th European Symposium on Research in Computer Security (ESORICS'13) (Lecture Notes in Computer Science)*.

Jeremy Dawson and Alwen Tiu. 2010. Automating open bisimulation checking for the spi-calculus. In *proceedings of IEEE Computer Security Foundations Symposium (CSF 2010)*.

Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. 2009. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security* 17, 4 (2009), 435–487.

doc 2013. Documentations of the code used for the 2013 Parliamentary election in Norway. https://brukerveiledning.valg.no/Dokumentasjon/Dokumentasjon/Forms/AllItems.aspx. (2013).

Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. 1992. A Practical Secret Voting Scheme for Large Scale Elections. In *AUSCRYPT'92: Workshop on the Theory and Application of Cryptographic Techniques (LNCS)*. Springer.

Kristian Gjøsteen. 2010. Analysis of an internet voting protocol. Cryptology ePrint Archive, Report 2010/380. (2010). http://eprint.iacr.org/.

Jens Groth. 2004. Evaluating Security of Voting Schemes in the Universal Composability Framework. In *2nd Int. Conference in Applied Cryptography and Network Security (ACNS 2004) (Lecture Notes in Computer Science)*, Vol. 3089. Springer, 46–60.

IACR. International association for cryptologic research. Elections page at http://www. iacr.org/elections/.

Ari Juels, Dario Catalano, and Markus Jakobsson. 2010. Coercion-Resistant Electronic Elections. In *Towards Trustworthy Elections: New Directions in Electronic Voting*. LNCS, Vol. 6000. Springer, 37–63.

Steve Kremer, Mark D. Ryan, and Ben Smyth. 2010. Election verifiability in electronic voting protocols. In *15th European Symposium on Research in Computer Security (ESORICS'10) (LNCS)*. Springer.

Ralf Küsters, Tomasz Truderung, and Andreas Vogt. 2010. A Game-Based Definition of Coercion-Resistance and its Applications. In *CSF'10: 23rd IEEE Computer Security Foundations Symposium*. IEEE Computer Society, 122–136.

R. Küsters, T. Truderung, and A. Vogt. 2012. Clash Attacks on the Verifiability of E-Voting Systems. In *33rd IEEE Symposium on Security and Privacy (S&P 2012)*. IEEE Computer Society, 395–409.

Helger Lipmaa. 2014. A Simple Cast-as-Intended E-Voting Protocol by Using Secure Smart Cards. Cryptology ePrint Archive, Report 2014/348. (2014). http://eprint.iacr.org/.

P.Y.A. Ryan, D. Bismark, J. Heather, S. Schneider, and Z. Xia. 2009. The Pret a Voter Verifiable Election System. *IEEE Transactions on Information Forensics and Security* (2009).

Ben Smyth. ProSwapper. http://www.bensmyth.com/proswapper.php.

Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J. Alex Halderman. 2014. Security Analysis of the Estonian Internet Voting System. In *Proc. 21st ACM Conference on Computer and Communications Security (CCS'14)*.