# Electronic voting: how logic can help

Véronique Cortier*

LORIA - CNRS, France

**Abstract.** Electronic voting should offer at least the same guarantees than traditional paper-based voting systems. In order to achieve this, electronic voting protocols make use of cryptographic primitives, as in the more traditional case of authentication or key exchange protocols. All these protocols are notoriously difficult to design and flaws may be found years after their first release. Formal models, such as process algebra, Horn clauses, or constraint systems, have been successfully applied to automatically analyze traditional protocols and discover flaws. Electronic voting protocols however significantly increase the difficulty of the analysis task. Indeed, they involve for example new and sophisticated cryptographic primitives, new dedicated security properties, and new execution structures.

After an introduction to electronic voting, we describe the current techniques for e-voting protocols analysis and review the key challenges towards a fully automated verification.

## 1 Context

Electronic voting promises a convenient and efficient way for collecting and tallying votes, avoiding human counting errors. Several countries now use electronic voting for politically binding elections. This is for example the case of Argentina, United States, Norway, Canada, or France. However electronic voting also causes controversy. Indeed these systems have been shown to be vulnerable to attacks. For example, the Diebold machines as well as the electronic machines used in India have been attacked [44, 58]. Consequently, some countries like Germany, Netherlands, or the United Kingdom have stopped electronic voting, at least momentarily [47].

Electronic voting covers two distinct families of voting systems: voting machines and Internet voting. Voting machines are computers placed at polling stations. They provide an interface for the voters to cast their vote and they process the ballots. Internet voting do not need physical polling stations: voters may simply vote using their own device (computers, smartphones, etc.) from home. In this paper we focus on Internet voting.

Internet voting raises several security challenges. Firstly, since votes need to be sent through the Internet, they obviously cannot be sent in clear. A simple solution would therefore to have all the voters encrypt their votes with the key of the voting server. At the end of the election, the server can then simply decrypt all the votes and announce the

* 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.

result. This is however not at all satisfactory since voters have no privacy with respect to the voting authority who could easily learn the vote of everyone. Moreover, such a solution offers no transparency: voters have no way to check that the announced outcome corresponds to the votes casted by voters. Therefore, two main security properties are put forward in the context of Internet voting: *confidentiality* and *verifiability*.

- Confidentiality or vote privacy ensures that no one can learn someone else' vote. Stronger than vote privacy is *receipt-freeness*: a voter should not be able to prove how she voted, even if she is willing to. In particular, she should not be given a receipt that can prove to a third-party for who she voted. This is to prevent vote buying for example. Even stronger than receipt-freeness is *coercion resistance*: even if a voter is temporarily under the control of a coercer, she should be able to cast the vote of her choice. This is typically achieved by letting voters re-vote (without the coercer being able to notice it).
- Verifiability ensures that anyone can check that the final result corresponds to the votes. In the literature, verifiability is typically split into sub-properties: *individual verifiability* states that a voter can check that her ballot appears on the bulletin board while *universal verifiability* states that the announced result corresponds to the ballots on the bulletin board. An additional property is *eligible verifiability*: only legitimate voters can vote, at most once. Of course, all these three properties are highly desirable.

A voting system should ensure all these properties despite the presence of attackers who may intercept communications and control some of the voters. Ideally, these properties should also hold when some voting authorities are corrupted too since it is desirable that voters can trust the result without having to trust all the authorities.

There is therefore a need for rigorous foundations for formalizing and reasoning about security of voting systems. In a slightly different context, formal methods have shown their usefulness in the analysis of security protocols. This line of research has started in the late 70's with the seminal work of Dolev and Yao [42] and Even and Goldreich [43]. Since then, many decision procedures have been developed to automatically analyse security properties such as authentication or confidentiality. Current leading tools include e.g. ProVerif [20, 21], Avispa [11], and Scyther [36]. They have been successfully applied to protocols of the literature as well as well-deployed protocols such as SSL and Kerberos, yielding the discoveries of flaws. A famous flaw is the "man-in-the-middle" attack found by Lowe [51] on the Needham-Schroeder asymmetric key protocol. More recently, an automated analysis [23] proved most of the secure tokens implementing the PKCS#11 standard to be broken. Similarly, a flaw was discovered using the Avispa tool on the Single-Sign-On protocol [12], used by many sites including Google.

Despite the similarities between standard security protocols and voting protocols, the current analysis techniques for standard protocols do not apply very well to voting systems. This is due to two main reasons. First, the cryptographic primitives used in e-voting are often ad-hoc and more complex than standard encryption and signatures. Second, privacy properties such as vote privacy or coercion-resistance are typically expressed as equivalence properties while the techniques developed so far mostly apply

to reachability properties. We survey here the particularities of Internet voting and describe the current limitations of existing techniques.

## 2 Existing systems for Internet voting

We first start by a short overview of some existing voting systems. This list is not meant to be exhaustive. Many systems used by companies are proprietary and there is few information available. We focus here on publicly available Internet voting systems, designed for achieving both privacy and verifiability.

**Helios [8]** is based on a protocol proposed by Cramers *et al* [35] with a variant proposed by Benaloh [16]. It has been used at the University of Louvain-la-Neuve to elect its president (recteur) and also in student elections. The IACR (International Association for Cryptologic Research) now uses Helios to elect its board, since 2010 [1]. It makes use of homomorphic encryption and zero-knowledge proofs. Helios has been proved to offer ballot privacy [33, 17, 18] (provided some fix is implemented [33]) and verifiability [32]. Several variants of Helios have then been proposed to enforce more properties. For example, [37] is a variant of Helios that guarantees *everlasting privacy*, that is, vote privacy is guaranteed even if the keys of the election get broken after the election. Belenios [32] offers better verifiability, in particular even if the election server is corrupted. However, Helios is not receipt-free (nor its variants): a voter may prove how she voted. It should therefore be used in low-coercion environment only.

**Civitas [29]** is one of the only implemented scheme (if not the only one) that offers both verifiability and coercion-resistance. It makes use of plaintext equivalence tests, re-encryption and mixnets. Civitas is still quite complex, both in terms of usability and computational complexity. It is therefore still unclear whether it is scalable to large elections.

**Norwegian protocol [46].** Norway has conducted Internet voting trials during the parliamentary election of 2013 and 2011. For the last election in 2013, about 250 000 voters of twelve selected municipalities were offered the possibility to use Internet voting [2]. The underlying voting protocol is developed by Scytl [3, 4] and is designed for both privacy and verifiability: voters are given a receipt that allow them to check that their vote has been counted, under some rather strong trust assumptions.

Several more academic voting protocols have been proposed in the literature such as the FOO protocol [45] or the Okamoto protocol [53].

## 3 Cryptographic primitives

Different formal models have been designed to reason about security protocols. Most popular ones include process algebra (e.g. CSP [55], applied-pi [5], spi-calculus [7]), strand spaces [56], Horn clauses [19], or constraint systems [52, 30]. They all have in common the fact that messages are represented by *terms*.

### 3.1 Terms

Given a signature $\mathcal{F}$, that is, a finite set of function symbols with their arity, given a set of variables $\mathcal{X}$, the set of terms $T(\mathcal{F}, \mathcal{X})$ is defined by the following grammar:

$$t, t_1, \ldots, t_n ::= x \mid f(t_1, \ldots, t_n) \qquad x \in \mathcal{X}$$

For example, a typical signature for security protocols is

$$\mathcal{F}_{\mathsf{enc}} = \{\mathsf{enc}, \mathsf{dec}, \mathsf{pair}, \mathsf{proj}_1, \mathsf{proj}_2\}$$

The function symbol $\mathsf{enc}$ represents encryption with associated decryption operator $\mathsf{dec}$ while $\mathsf{pair}$ represents concatenation with associated projectors $\mathsf{proj}_1$ and $\mathsf{proj}_2$. The properties of the primitives are then expressed through an *equational theory*. For (symmetric) encryption and concatenation, the usual equations are the following ones:

$$\begin{gathered}\mathsf{dec}(\mathsf{enc}(x, y), y) = x \\ \mathsf{proj}_1(\mathsf{pair}(x, y)) = x \\ \mathsf{proj}_2(\mathsf{pair}(x, y)) = y\end{gathered}$$

For example, $\mathsf{proj}_2(\mathsf{dec}(\mathsf{enc}(\mathsf{pair}(a, n), k), k) = n$.

The equational theories are rather simple and belong to the class of *subterm convergent theories* [6]: they are convergent and the right member of an equational is always a subterm of the left member, or a constant. Deciding secrecy or authentication properties have been shown to be decidable both for passive [6] and active adversaries [15], for a bounded number of sessions. Some tools such as ProVerif [20] or Akiss [25] can handle arbitrary theories (with no termination guarantee of course) and typically behave well for subterm convergent theories.

### 3.2 Equational theories for e-voting.

Cryptographic primitives for e-voting systems are however more complex than standard primitives such as encryption or signatures. We review here some examples.

A rather standard primitive is blind signature, used for example in the FOO protocol [45]. While signatures are typically designed to be non malleable, blind signatures support some form of malleability. In FOO, voters send a blinded version of their vote to the voting authority, get it signed and then retrieve the signature of the authority on their (unblinded) vote. This property can be formalised as follows [41]:

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

This equation means intuitively that knowing the blinding factor and the signature of a blinded message, anyone can compute the signature of the original message.

Another example comes from the Helios protocol described in Section 2. This protocol involves homomorphic encryption, that is, the combination of two encrypted votes yields the encryption of the sum of the votes. This property is at the heart of the Helios

$$\mathsf{proj}_1(\mathsf{pair}(x, y)) = x \tag{1}$$
$$\mathsf{proj}_2(\mathsf{pair}(x, y)) = y \tag{2}$$
$$\mathsf{dec}(\mathsf{aenc}(x_{plain}, x_{rand}, \mathsf{pk}(x_{sk})), x_{sk}) = x_{plain} \tag{3}$$
$$\mathsf{dec}(\mathsf{blind}(\mathsf{aenc}(x_{plain}, x_{rand}, \mathsf{pk}(x_{sk})), x_{blind}), x_{sk}) = \mathsf{blind}(x_{plain}, x_{blind}) \tag{4}$$
$$\mathsf{aenc}(x_{pl}, x_{rand}, x_{pub}) \circ \mathsf{aenc}(y_{pl}, y_{rand}, x_{pub}) = \mathsf{aenc}(x_{pl} \diamond y_{pl}, x_{rand} * y_{rand}, x_{pub}) \tag{5}$$
$$\mathsf{renc}(\mathsf{aenc}(x_{plain}, x_{rand}, \mathsf{pk}(x_{sk})), y_{sk}) = \mathsf{aenc}(x_{plain}, x_{rand}, \mathsf{pk}(x_{sk} + y_{sk})) \tag{6}$$
$$\mathsf{unblind}(\mathsf{blind}(x_{plain}, x_{blind}), x_{blind}) = x_{plain} \tag{7}$$
$$\mathsf{Checksign}(x_{plain}, \mathsf{vk}(x_{id}), \mathsf{sign}(x_{plain}, x_{id})) = \mathsf{ok} \tag{8}$$
$$\mathsf{Checkpfk}(\mathsf{vk}(x_{id}), \mathsf{ball}, \mathsf{pfk}(x_{id}, x_{rand}, x_{plain}, \mathsf{ball})) = \mathsf{ok} \quad \text{where } \mathsf{ball} = \mathsf{aenc}(x_{plain}, x_{rand}, x_{pub}) \tag{9}$$
$$\mathsf{Checkpfkk}(\mathsf{vk}(x_{id}), \mathsf{ball}, \mathsf{pfkk}(\mathsf{vk}(x_{id}), x_{bk}, x_{plain}, \mathsf{ball})) = \mathsf{ok} \quad \text{where } \mathsf{ball} = \mathsf{renc}(x_{plain}, x_{bk}) \text{ or } \mathsf{ball} = \mathsf{blind}(x_{plain}, x_{bk}) \tag{10}$$

The symbols $+$, $*$, $\diamond$, and $\circ$ are assumed to be commutative and associative.

**Fig. 1.** Equations used in [34] to model the protocol used in Norway.

protocol since anyone can combine the votes to obtain the result (in an encrypted form). This homomorphic property can be expressed by the following equation:

$$\mathsf{aenc}(v_1, r_1, \mathsf{pk}) * \mathsf{aenc}(v_2, r_2, \mathsf{pk}) = \mathsf{aenc}(v_1 + v_2, r_1.r_2, \mathsf{pk})$$

where $*$, $+$, and . are associative and commutative functional symbols. Note that aenc is a ternary symbol that represents (randomized) asymmetric encryption. The second argument $r$ represents the randomness used for encrypting. Using randomized encryption is crucial in e-voting to prevent an attacker to compare encrypted votes. However, associativity and commutativity are typically not supported by existing tools for security protocols.

Other examples of primitives used in e-voting are trapdoor commitments schemes, zero-knowledge proofs, designated verifier zero-knowledge proofs, or plaintext equivalence tests. Of course, voting systems may mix several of those primitives. For the sake of illustration, we display in Figure 1 the complete equational theory used in [34] to model the protocol used in Norway. It is clearly out of reach of existing tools.

## 4 Security properties

Most existing techniques developed so far for security protocols focus on reachability properties, that is, properties of the form: "for any execution trace, nothing bad happens". Confidentiality of keys or nonces as well as authentication properties are typical security properties that fall into the category of reachability properties. Ballot secrecy is

however not expressed as a reachability property. Indeed, ballot privacy does not mean that the *value* of the vote remains secret. On the contrary, all the possible values of a vote (for example 0 or 1 in case of a referendum) are well-known by anyone. Therefore ballot secrecy is typically stated as an indistinguishability property [41]: an attacker should not notice any difference when Alice is voting 0 and Bob is voting 1 from the converse scenario where votes are swapped (Alice votes 1 and Bob votes 0). This can be easily expressed in process algebra calculus that has a notion of behavioral equivalence $\approx$.

$$V_{\mathsf{Alice}}(0) \mid V_{\mathsf{Bob}}(1) \approx V_{\mathsf{Alice}}(1) \mid V_{\mathsf{Bob}}(0)$$

where the process $V_\alpha$ represents voter $\alpha$.

Coercion-resistance and receipt-freeness are also stated using equivalence properties [40].

ProVerif is one of the only tools that can check equivalence properties. It actually tries to prove a stronger property than behavioral equivalence [21] for couple of protocols that have a very similar structure. However ProVerif does not work very well on vote privacy, although it has recently improved [27]. Several recent (and preliminary) tools have been proposed to check equivalence of protocols, for a bounded number of sessions. AKiSs [25] can check (trace) equivalence for arbitrary (convergent) theories but is not guaranteed to terminate. APTE [26] checks (trace) equivalence for a large family of standard primitives (encryption, signatures, hashes, concatenation) and can handle non determinism and else branches. SPEC [38] implements a procedure for open bisimulation, a notion of equivalence stronger than the standard notion of trace equivalence.

Verifiability has not yet reached the same level of maturity than ballot privacy in terms of modeling. A first proposal has been made in [49] that provides formal definitions of both individual, universal, and eligibility verifiability. A much simpler yet probably weaker definition [48, 32] states that the final outcome should contain:

- the votes of all voters that have voted and performed appropriate checks;
- a subset of the votes of the voters that did vote but did not check anything;
- at most $k$ arbitrary valid votes where $k$ is the number of voters under the control of the attacker.

Another approach [50] proposes a very general framework to define verifiability and also *accountability*, a notion that captures that a system should not only be verifiable but in case something wrong happened, it should be possible to blame who misbehaved. Due to its generality, the approach developed in [50] does not provide with a unique definition of verifiability. Instead one has to instantiate the framework for each voting system.

It is likely that new alternative definitions will still emerge to formally define verifiability.

## 5 Conclusion

Voting systems raise challenging issues to the area of formal verification of security protocols. First, tools and techniques need to shift from reachability to equivalence-

based properties. The interest of equivalence-based properties is not confined to voting systems. Indeed behavioural equivalences are used more generally to formalize privacy properties such as anonymity or unlinkability in many different contexts (RFIDs [24, 10], passports [28], mobile telephony systems[9]). They may also express security properties closer to game-based definitions used in cryptography. For example, learning even a single bit of a key is considered as an attack in cryptography. The fact that not even a bit of the secret shall be linked is called *strong secrecy* in symbolic models and is defined through the equivalence of two processes. More generally, game-based cryptographic definitions can be defined in symbolic models through equivalences [31, 39]. New tools have been designed to automatically check equivalence of security protocols, for a bounded number of sessions. This is in particular the case of AKiSs [25], APTE [26], and SPEC [38]. For an unbounded number of sessions, the only available tool is ProVerif [20, 21] which can check equivalence for pairs of protocols that have the same structure and for reasonably general equational theories.

Another major issue of e-voting systems is the complexity and variety of cryptographic primitives that include homomorphic encryption, re-encryption mixnets, zero-knowledge proofs, and trapdoor commitments. These primitives may be formalized through equational theories. However, most of them include associative and commutative symbols and are out of reach of existing tools, even for reachability properties.

Moreover, the primitives used in e-voting challenge the abstractions made in symbolic models: although the resulting equational theories are already quite complex, some equations may still be missed. In cryptography, more accurate models are used: instead of using process algebra with terms, protocols and attackers are simply any (polynomial) Turing machines. While cryptographic and symbolic models largely differ, symbolic models were shown to be *sound* with respect to cryptographic ones, that is, any protocol proved to be secure in symbolic models is deemed secure in cryptographic ones. Such a soundness result holds for most standard primitives [13, 22] but very few results exist outside these standard primitives ([54] being one of the few exceptions). Some primitives like the Exclusive Or were even shown to be impossible to soundly abstract [57]. It may be therefore preferable in some cases to analyse e-voting protocols directly in cryptographic models, possibly using recently developed techniques that assist and partially automate the proof (see for example the line of research developed on EasyCrypt [14]).

To conclude, we expect e-voting to continue to foster the development of new techniques and tools in both symbolic and cryptographic approaches.

## References


1. International association for cryptologic research. Elections page at http://www.iacr.org/elections/.
2. Web page of the Norwegian government on the deployment of e-voting. `http://www.regjeringen.no/en/dep/krd/prosjekter/e-vote-2011-project.html`.
3. Documentations of the code used for the 2013 parlementary election in Norway. `https://brukerveiledning.valg.no/Dokumentasjon/Dokumentasjon/Forms/AllItems.aspx`, 2013.

4. KRD - evalg2011 platform - update for 2013 parliamentary elections. `https://brukerveiledning.valg.no/Dokumentasjon/Dokumentasjon/Norway-2013_BulletinBoard_v1.2.pdf`, 2013.
5. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *28th ACM Symposium on Principles of Programming Languages (POPL'01)*, 2001.
6. Martín Abadi and Véronique Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 367(1–2):2–32, 2006.
7. Martín Abadi and Andrew D. Gordon. A Calculus for Cryptographic Protocols: The Spi Calculus. In *CCS'97: 4th ACM Conference on Computer and Communications Security*, pages 36–47. ACM Press, 1997.
8. Ben Adida, Olivier de Marneffe, Oliver Pereira, and Jean-Jacques Quisquater. 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*, 2009.
9. M. Arapinis, L. Mancini, E. Ritter, and M. Ryan. Privacy through pseudonymity in mobile telephony systems. In *21st Annual Network and Distributed System Security Symposium (NDSS'14)*, 2014.
10. Myrto Arapinis, Tom Chothia, Eike Ritter, and Mark Ryan. Analysing Unlinkability and Anonymity Using the Applied Pi Calculus. In *CSF'10: 23rd Computer Security Foundations Symposium*, pages 107–121. IEEE Computer Society, 2010.
11. 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. The AVISPA Tool for the automated validation of internet security protocols and applications. In *17th International Conference on Computer Aided Verification, CAV'2005*, volume 3576 of *Lecture Notes in Computer Science*, pages 281–285. Springer, 2005.
12. Alessandro Armando, Roberto Carbone, Luca Compagna, Jorge Cuellar, and Llanos Tobarra Abad. 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)*, pages 1–10, 2008.
13. Michael Backes and Birgit Pfitzmann. Symmetric encryption in a simulatable Dolev-Yao style cryptographic library. In *Proc. 17th IEEE Computer Science Foundations Workshop (CSFW'04)*, pages 204–218, 2004.
14. G. Barthe, B. Grégoire, S. Heraud, and S. Zanella-Béguelin. Computer-aided security proofs for the working cryptographer. In *Advances in cryptology – (Crypto'11)*, volume 6841 of *Lecture notes in computer science*, page 71–90. Springer, 2011.
15. Mathieu Baudet. Deciding security of protocols against off-line guessing attacks. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*, pages 16–25. ACM Press, November 2005.
16. Josh Benaloh. Ballot casting assurance via voter-initiated poll station auditing. In *Proceedings of the Second Usenix/ACCURATE Electronic Voting Technology Workshop*, 2007.
17. David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting Helios for provable ballot secrecy. In *Proceedings of the 16th European Symposium on Research in Computer Security (ESORICS'11)*, volume 6879 of *Lecture Notes in Computer Science*, 2011.
18. David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to helios. In *Advances in Cryptology - (ASIACRYPT 2012)*, pages 626–643, 2012.
19. B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proc. of the 14th Computer Security Foundations Workshop (CSFW'01)*. IEEE Computer Society Press, June 2001.

20. Bruno Blanchet. An automatic security protocol verifier based on resolution theorem proving (invited tutorial). In *20th International Conference on Automated Deduction (CADE-20)*, July 2005.
21. Bruno Blanchet, Martín Abadi, and Cédric Fournet. Automated verification of selected equivalences for security protocols. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005)*, pages 331–340. IEEE Computer Society, June 2005.
22. Florian Böhl, Véronique Cortier, and Bogdan Warinschi. Deduction soundness: Prove one, get five for free. In *20th ACM Conference on Computer and Communications Security (CCS'13)*, Berlin, Germany, 2013.
23. Matteo Bortolozzo, Matteo Centenaro, Riccardo Focardi, and Graham Steel. Attacking and fixing PKCS#11 security tokens. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS'10)*, pages 260–269. ACM Press, October 2010.
24. Mayla Brusó, Konstantinos Chatzikokolakis, and Jerry den Hartog. Formal verification of privacy for RFID systems. In *CSF'10: 23rd Computer Security Foundations Symposium*, pages 75–88. IEEE Computer Society, 2010.
25. Rohit Chadha, Ştefan Ciobâcă, and Steve Kremer. Automated verification of equivalence properties of cryptographic protocols. In *21th European Symposium on Programming (ESOP'12)*, LNCS. Springer, 2012. To appear.
26. Vincent Cheval. 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, April 2014.
27. Vincent Cheval and Bruno Blanchet. Proving more observational equivalences with ProVerif. In *Proceedings of the 2nd International Conference on Principles of Security and Trust (POST'13)*, Lecture Notes in Computer Science, pages 226–246. Springer, March 2013.
28. Vincent Cheval, Véronique Cortier, and Antoine Plet. 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)*, volume 8043 of *Lecture Notes in Computer Science*, pages 708–723. Springer, July 2013.
29. Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a secure voting system. In *Proc. IEEE Symposium on Security and Privacy*, pages 354–368, 2008.
30. H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of Exclusive Or. In *Proc. of 18th Annual IEEE Symposium on Logic in Computer Science (LICS '03)*, pages 271–280. IEEE Computer Society, 2003.
31. Hubert Comon-Lundh and Véronique Cortier. Computational soundness of observational equivalence. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*, pages 109–118, Alexandria, Virginia, USA, October 2008. ACM Press.
32. Véronique Cortier, David Galindo, Stephane Glondu, and Malika Izabachene. A generic construction for voting correctness at minimum cost - application to helios. Cryptology ePrint Archive, Report 2013/177, 2013.
33. Véronique Cortier and Ben Smyth. Attacking and fixing helios: An analysis of ballot secrecy. *Journal of Computer Security*, 21(1):89–148, 2013.
34. Véronique Cortier and Cyrille Wiedling. A formal analysis of the norwegian e-voting protocol. In *Proceedings of the 1st International Conference on Principles of Security and Trust (POST'12)*, volume 7215 of *Lecture Notes in Computer Science*, pages 109–128. Springer, March 2012.
35. Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'97)*, pages 103–118, 1997.
36. Cas Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, USA,*

*Proc.*, volume 5123/2008 of *Lecture Notes in Computer Science*, pages 414–418. Springer, 2008.
37. Edouard Cuvelier, Olivier Pereira, and Thomas Peters. 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, 2013.
38. Jeremy Dawson and Alwen Tiu. Automating open bisimulation checking for the spi-calculus. In *proceedings of IEEE Computer Security Foundations Symposium (CSF 2010)*, 2010.
39. Stéphanie Delaune, Steve Kremer, and Olivier Pereira. Simulation based security in the applied pi calculus. In *Proceedings of the 29th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'09)*, volume 4 of *Leibniz International Proceedings in Informatics*, pages 169–180, December 2009.
40. Stéphanie Delaune, Steve Kremer, and Mark Ryan. Coercion-Resistance and Receipt-Freeness in Electronic Voting. In *CSFW'06: 19th Computer Security Foundations Workshop*, pages 28–42. IEEE Computer Society, 2006.
41. Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, 2009.
42. D. Dolev and A.C. Yao. On the security of public key protocols. In *Proc. of the 22nd Symp. on Foundations of ComputerScience*, pages 350–357. IEEE Computer Society Press, 1981.
43. S. Even and O. Goldreich. On the security of multi-party ping-pong protocols. In *Technical Report*. IEEE Computer Society Press, 1983.
44. Ariel J. Feldman, J. Alex Halderman, and Edward W. Felten. Security analysis of the diebold accuvote-ts voting machine. `http://itpolicy.princeton.edu/voting/`, 2006.
45. Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In *AUSCRYPT'92: Workshop on the Theory and Application of Cryptographic Techniques*, LNCS. Springer, 1992.
46. Kristian Gjøsteen. Analysis of an internet voting protocol. Cryptology ePrint Archive, Report 2010/380, 2010. `http://eprint.iacr.org/`.
47. Jordi Barrat i Esteve, Ben Goldsmith, and John Turner. International experience with e-voting. Technical report, Norwegian E-Vote Project, 2012.
48. Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-Resistant Electronic Elections. In *Towards Trustworthy Elections: New Directions in Electronic Voting*, volume 6000 of *LNCS*, pages 37–63. Springer, 2010.
49. Steve Kremer, Mark D. Ryan, and Ben Smyth. Election verifiability in electronic voting protocols. In *15th European Symposium on Research in Computer Security (ESORICS'10)*, LNCS. Springer, 2010.
50. R. Küsters, T. Truderung, and A. Vogt. Clash Attacks on the Verifiability of E-Voting Systems. In *33rd IEEE Symposium on Security and Privacy (S&P 2012)*, pages 395–409. IEEE Computer Society, 2012.
51. G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS'96)*, volume 1055 of *LNCS*, pages 147–166. Springer-Verlag, march 1996.
52. J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proc. of the 8th ACM Conference on Computer and Communications Security (CCS'01)*, 2001.
53. Tatsuaki Okamoto. Receipt-Free Electronic Voting Schemes for Large Scale Elections. In *5th International Workshop on Security Protocols (SP'97)*, volume 1361 of *LNCS*, pages 25–35. Springer, 1998.
54. Hideki Sakurada. Computational soundness of symbolic blind signatures under active attacker. In *6th International Symposium on Foundations and Practice of Security (FPS'13)*, pages 247–263, 2013.

55. S. Schneider. Verifying authentication protocols with CSP. In *Proc. of the 10th Computer Security Foundations Workshop (CSFW'97)*. IEEE Computer Society Press, 1997.
56. J. Thayer, J. Herzog, and J. Guttman. Strand spaces: proving security protocols correct. *IEEE Journal of Computer Security*, 7:191–230, 1999.
57. Dominique Unruh. The impossibility of computationally sound xor, July 2010. Preprint on IACR ePrint 2010/389.
58. Scott Wolchok, Eric Wustrow, J. Alex Halderman, Hari K. Prasad, Arun Kankipati, Sai Krishna Sakhamuri, Vasavya Yagati, and Rop Gonggrijp. Security analysis of india's electronic voting machines. In *17th ACM Conference on Computer and Communications Security (CCS 2010)*, 2010.