SOFTWARE SECURITY EVALUATION FOR EVOTING SYSTEMS

Domenica Bagnato[1]

DOI: 10.24989/ocg.v341.31

Abstract
This paper follows on from Recommendation CM/Rec(2017)5 of the Council of Europe on eVoting in defining the basic principles and requirements of a software security evaluation according to Common Criteria that distinguishes the main differences in the two recommendations, namely verifiability and voter secrecy. It proposes a definition for assets, threats and objectives and discusses the differences between the current Recommendation CM/Rec(2017)5 and its predecessor, Rec(2004)11.

1. Introduction

Any eVoting protocol and its implementation has to be verified in a formal procedure as is customary with security sensitive software. The de facto standard for software security evaluation is Common Criteria (CC) [1]. CC is available on several "evaluation assurance levels, EAL" of increasing stringency, starting from Level 1, which basically only assures the functioning of the software, to Level 7, which typically applies to military-grade systems. Each level defines a set of formal requirements for, most importantly, defining the threats and measures used to protect the software from the threats. Some examples of EAL certifications may be Microsoft SQL Server 2019 (EAL 2+), IBM z/OS Version 2 (EAL 4+) or the Malayan digital ID (EAL 3+) [2]. The symbol, "+", in this context means that requirements from higher levels are included in the main level ("augmented"). For example, EAL 2+ may mean that the software includes assurance level 2 and some parts from Levels 3 and 4.

The general procedure is to define a Protection Profile for a class of products and then certify a product against the Protection Profile (PP) [3]. The PP is therefore the blueprint which the product itself is certified against. The EAL of the PP therefore also determines the EAL of the certified product. Rec(2004)11 on eVoting [4] was an attempt at providing a set of standards for eVoting systems, whereby "eVoting" was understood in its broadest form and also encompassed on-site voting machines, kiosks and other systems. In this paper, the term eVoting will only refer to Internet-based remote voting. Where other forms of electronic vote casting are meant, special reference will be made. Rec(2004)11, however, did not just contend itself with defining security and quality standards of eVoting systems. In Appendix III, Section F, it details the certification requirements and deals with the topic in a terminology very close to the methodology of CC. In particular,

(i) It defines the assets to be protected;
(ii) the threats to be countered; and
(iii) the objectives to counter the threats.

[1] Domenica Bagnato, Hierodiction Software GmbH. Email: domenica.bagnato@hierodiction.com

This description would have readily lent itself to the definition of a CC PP, which could have then been certified and used as a yardstick for product certification. Unfortunately, such a profile never materialised and therefore "recommendation-compliance" as maintained by some vendors remains a loose term, which is regrettable in a field where trust is of upmost importance. [5, 6]

Rec(2004)11 contained shortcomings, which became clear in the following years as a result of a number of eVoting applications [7]. The main shortcomings manifested themselves in the protection of voting secrecy and auditability. This led to the vastly improved Recommendation CM/Rec(2017)5, which defines very stringent requirements in terms of voting secrecy and reproducibility/audit[2]. CM/Rec(2017)5 made significant progress in terms of the requirements but was a step back in terms of defining a certification process for it abandoned the approach shown in Rec(2004)11, Annex II, Section F and does not include any steps towards establishing a security evaluation. This may be due to the fact that the initial move towards establishing such a security profile as incorporated in Rec(2004)11, II.F never came to fruition. However, in the absence of such a formal profile incorporating the Standards listed in the Recommendation, any independent evaluation would be ad-hoc and cannot be based on a formal and standardised framework, which implements the intent of the Recommendation. The result is that there is no comparable and standard, formalised method of assessing eVoting systems.

Hence, it has to be noted that content-wise CM/Rec(2017)5 [9] provides a considerable improvement in the standard definition of a much tighter framework for what an eVoting protocol, and implementation, must provide, but unfortunately, it is also less stringent in its formal path towards evaluation than Rec(2004)11.

This paper provides steps towards defining the Assets and Threats according to CM/Rec (2017)5 and discusses security objectives to achieve them. The special focus will be on Internet-based eVoting and the additional Standards (=requirements) of the upgraded Recommendation. Before that, let us briefly look at the main improvements incorporated in CM/Rec(2017)5 as compared to its predecessor.

2. The Decisive Improvements in CM/Rec(2017)5

CM/Rec(2017)5 provides two main improvements, namely: (i) a better and more explicit protection of voter secrecy; and (ii) verification. The latter is subsumed under Section "Free Suffrage", but in fact provides for strict verifiability of the vote. The Recommendation works with "Standards" that are required to be fulfilled, and the standards listed below are the main standards pertaining to these two improvements (the standard number is included at the beginning, my emphasis added).

<u>Verification (free suffrage) [9, Section III]</u>

11. It shall be ensured that the e-voting system presents an ***authentic ballot*** *and authentic information to the voter.*

15. The voter shall be able to ***verify that his or her intention is accurately represented*** *in the vote and that the* ***sealed vote has entered the electronic ballot box without being altered****. Any undue influence that has modified the vote shall be detectable.*

[2] For details see [8].

16. The voter shall receive ***confirmation*** *by the system that the* ***vote has been cast*** *successfully and that the whole voting procedure has been completed.*

17. The e-voting system shall provide sound evidence that ***each authentic vote is accurately included in the respective election results****. The evidence should be verifiable by means that are* ***independent*** *from the e-voting system.*

18. The system shall provide sound evidence that ***only eligible voters' votes*** *have been* ***included*** *in the respective final* ***result****. The evidence should be verifiable by means that are independent from the e-voting system.*

The key Standards here are clearly 15 (individual verifiability) and 17 (general verifiability). Please note that individual verifiability as defined in Standard 15 only reaches to the ballot box[3], however general verifiability in Standard 17 reaches to the election result.

Note that standards 17 and 18 require independent means of verification. The verification requirements are largely new to CM/Rec(2017)5 and go well beyond the general auditability requirements in Rec(2004)11.

Voting Secrecy [9, Section IV]

21. The e-voting system and any authorised party shall ***protect authentication data*** *so that unauthorised parties cannot misuse, intercept, modify, or otherwise gain knowledge of this data.*

23. An e-voting system ***shall not provide the voter with proof of the content of the vote*** *cast for use by third parties.*

24. The e-voting system ***shall not allow the disclosure*** *to anyone of the number of votes cast for any voting option* ***until after the closure of the electronic ballot box****. This information shall not be disclosed to the public until after the end of the voting period.*

25. E-voting shall ensure that the ***secrecy of previous choices*** *recorded and erased by the voter before issuing his or her final vote is respected.*

26. The e-voting process, in particular the counting stage, shall be organised in such a way that it is ***not possible to reconstruct a link between the unsealed vote and the voter.*** *Votes are, and remain, anonymous.*

Standard 21 is mainly a standard provision of protecting authentication data, however it is certainly a crucial property of any eVoting system.

Standard 23 is the real (hidden) challenge in CM/Rec(2017)5 in remote eVoting. It needs to be understood in context of individual, and to some extent general, verifiability and at the same of time strict protection of voting secrecy (see Standard 26 below).[4] The implementation guidelines of the Recommendation [11] provide some rather detailed information for onsite electronic voting, for

[3] This need not necessarily be so, for a voting system where individual verifiability reaches to the election result without compromising voting secrecy, cf [8].
[4] For more discussion on these limitations, cf [8]

remote eVoting they limit themselves to such remarks as "*the voter should be informed of possible risks to voting secrecy*" (Remark c) and "*how to delete, where it is possible, traces of the vote from the device used to cast the vote."* (Remark d). The essential friction between no-proof verifiability and voting secrecy is not addressed. However, depending on the eVoting protocol used, in remote eVoting it may turn out to be *the* challenge to eVoting in many respects in addressing the core of the eVoting issue in general that is how to be verifiable and protect secrecy at the same time.

Standard 24 protects from manipulative premature disclosures of election results during an ongoing election to achieve (de)mobilisation effects.

The core here is clearly Standard 26, which is worded in much stronger terms than the corresponding standard in Rec(2004)11. Standard 25 applies to voting regimes, where multiple replacement votes are possible, such as in Estonia. [12]

In the opinion of the author these points represent the core improvements of the new Recommendation as compared to Rec(2004)11. The remaining sections of the paper will propose a representation and inclusion in a CC-oriented style following Rec(2004)11, Annex III, Section F that will hopefully trigger a discussion in this direction. Generally speaking, the paper advances the hypothesis that only a certified PP incorporating those points will enable eVoting system verification and lead to credible and trusted systems. The following proposal could be a first step in this direction.

A CC Protection Profile must inherently be product- and system-independent, however, in some cases, reference will be made to the two main eVoting protocol families today, the Envelope and the Token protocol family that show vastly different security properties in key areas discussed in this paper. For more details, see [8].

3. Assets

The representation below follows the same logic as Rec(2004)11, Annex III, Section F in distinguishing between assets pertaining to the different stages of eVoting (pre-voting, vote casting and post voting stages).

3.1. Assets Pertaining to Verification

The following table lists the Protection Profile Assets resulting from the above standards with the standard number added to the asset description (SDxx). Multiple assignments are possible.

		Stage		
Asset ID	**Asset**	**Pre**	**Cast**	**Post**
A.AuthBallot	*Authentic ballot is presented to the voter (SD11)*		*x*	
A.AuthClient	Authenticity of the voting client (SD11)	x	x	x
A.IndivVerify	Individual verifiability reaching to the ballot box (SD15, 16)	x	x	x
A.ResultVerify	Verifiability of the end result (SD17, 18)	x	x	x

Table 1: Assets in verification (Pre-voting only applies to the token protocol family)

The A.AuthBallot asset in table 1 is a subset of the A.AuthClient asset, which encompasses all information created by the eVoting system including the operational parameters and specifically the voting period (Envelope protocol) or registration and voting period (token protocol) in the two-stage protocols (cf. [13]). However, it is best to use the generalised A.AuthClient and not the

A.AuthBallot so as to eliminate redundancy. From the explanatory memorandum it is not quite clear, whether it encompasses the audit data produced by the eVoting system and if so, to what extent. We will assume that this asset does not include them to avoid duplication with A.ResultVerify (see below).

As for A.IndivVerify, Standards 15 and 16 arguably aim to assure the voter that he/she actually cast the vote that was intended. In this context, Standard 15 can be seen as a superset of Standard 16. It should be noted that ex post verification starts in the registration phase or voting phase depending on the protocol, but definitely not in the post voting stage. This applies to digital signatures or other authentication information provided in casting a vote in the Envelope protocol system [14] or the issue of a token in the pre-voting stage for a Token protocol system [15]. In any case, it should be understood, that verification is not something added right at the end of the voting process.

Furthermore, Standards 17 and 18 combined represent A.ResultVerify that is the general verification of the result, with Standard 18 being the subset of Standard 17. If the result is independently verified in its entirety, that necessarily includes verification that each vote cast and counted is submitted by an eligible voter, who also voted only once. This also includes all audit trails and logging.

3.2. Assets Pertaining to Voter Secrecy

Table 2 lists the relevant Assets.

		Stage		
Asset ID	**Asset**	**Pre**	**Cast**	**Post**
A.AuthData	Protect authentication data (SD21)	x	x	
A.NoProof	Voter not provided with a proof of how he/she voted (SD23)		x	x
A.NoPremature	No premature disclosure of election results (SD24)			x
A.Secrecy	Voting secrecy including replaced choices (SD25, 26)		x	x

Table 2: Assets in voting secrecy

As for A.AuthData, the stage assignment heavily depends on the type of protocol. Token protocols would have to protect it in the pre-voting phase only, unless the token is considered authentication data in the vote casting stage. We will operate under this assumption. In an Envelope protocol, the authentication data, including the digital signature on the sealed vote envelope, is needed in the vote casting stage. We will hence generally assume that A.AuthData needs to be protected in both stages, understanding that for the Token protocol, the authentication data in the vote casting stage is the token.

A.NoProof in itself is rather straightforward. It gains its consequences from the linkage to A.Secrecy and A.IndivVerify (and to some extent A.ResultVerify). A.NoPremature has to be qualified for practical reasons. There is a coalition of actors, such as the election committee in its entirety, that can actually violate this asset. That is where technical safeguards end and organisational precautions must start.

Standards 25 and 26 ensure voting secrecy, with 25 being applicable to electoral systems with replacement votes. In addition, here, there is a factual difference between Envelope and Token protocols in that the former stores digital voter signature and sealed vote together, whereas the latter stores the vote together with the authenticated token. The further discussion of threats and objectives must take that difference into account.

4. Threats

Table 3 below maps the Threats identified to the Assets above. We will discuss each of the Threats below.

Threat / Asset	A.Auth Client	A.Indiv Verify	A.Result Verify	A.Auth Data	A.No Proof	A.No Premature	A. Secrecy
T.ClientForgery	x						
T.BallotForgery	x						
T.AuditForgery			x	x		x	
T.AuthentDataForgery		x	x			x	x
T.OwnVoteMisrepresent		x	x				
T.VoteModify			x				
T.VoteMultiple			x	x			
T.PhantomVoters			x	x			
T.InsertPhantomVotes			x				
T.UnlawfulResultAccess						x	
T.Miscount			x				
T.NoIndependentRecount			x				
T.LinkVoterVote							x
T.TimeManipulation	x					x	

Table 3: Threats

T.ClientForgery involves the forgery of the entire voting/registration client, in most cases a Java applet. T.BallotForgery involves presentation of a manipulated list of candidates including their order, or the wrong assignment of preferential votes to the main voting options.

T.AuditForgery either involves manipulated new log entries, manipulation of existing entries or no entry in cases, where there should be a log entry. As can be seen from Table 3, it can be directed against a number of Assets. A premature disclosure of results, for instance, would also require to “switch off” the relevant logging for the disclosure to go permanently undetected.

T.AuthentDataForgery covers the forgery of any means of voter ID and authentication used including voting tokens in a Token protocol. It constitutes a massive attack on general verification and sometimes would go undetected in maintaining the secrecy of the vote. For example, an individual may request individual vote verification of another voter using fabricated authentication data.

T.OwnVoteMisrepresent concerns the presentation of the original choice to the voter even though the true vote, stored in the system, has been modified. This then of course also entails an attack on the general verification. T.VoteModify refers to the modification of the vote after it was cast (and verified by the voter). Both need to be conceptually distinguished. It is possible to represent the correct vote (as cast) to the verification-seeking voter and yet misrepresent it in the general tally.

T.VoteMultiple represents the cast of multiple votes by the voter him/herself, where this is not part of the election procedure (and the old votes are invalidated). This of course corrupts general verifiability, but also constitutes an attack on the authentication process, which should ensure uniqueness.

T.PhantomVoters constitutes the insertion of bogus voters in the roll, for whom one may then cast - formally correct - votes. It therefore primarily attacks authentication. T.InsertPhantomVotes means to insert votes that were never cast by somebody who as such is a real and valid voter. T.OwnVoteMisrepresent to T.InsertPhantomVotes form a bundle of threats that also require similar countermeasures in the objectives.

T.UnlawfulResultAccess constitutes the unlawful access to the election result with two caveats: (i) the threat attacks Asset A.NoPremature, however it goes well beyond this; (ii) the technical barriers must be overridable by a decision of the election committee and the technical safeguards cannot possibly counteract a combined collusive act by the election committee.

T.Miscount involves a number of possibilities. To see them, the term “vote counting” needs to be understood. The counting process does not encompass merely a simple vote count, eg, number of votes per party. The threat therefore involves the following elements:

(i) No safeguards against votes that are not authentic and unchanged;
(ii) No safeguards against voters casting multiple votes (replaced votes notwithstanding)

This goes well beyond counting how many votes were cast for option “X”.

T.NoIndependentRecount covers the threat that no independent recount outside the eVoting system is possible or that the recount is reduced to a simple vote count without the authenticity checks in the original count (see T.Miscount).

T.LinkVoterVote covers the threat that a link may be provided between the filled-in ballot and the authentication data of the voter. Of course, this is diametrically opposed to the verification requirements, where it must be established that the vote was cast by an (any) authentic voter (and just once). This goal antinomy is the real crunch of eVoting protocol and system design and it was accentuated by the revised CM/Rec(2017)5.

T.TimeManipulation covers the threat that the time perceived by the voting client or server is wrong and this impairs voting integrity. The main issue here, not covered by the focus of this paper, is of course to prevent vote casting in providing an incorrect time, after voting closed, to the voting client that consequently rejects vote casting, or registration in a Token protocol. Additionally, within the focus of this paper, system time manipulation plays a role in attacking Asset A.NoPremature enabling unlawful access to the interim results.

5. Security Objectives

Following CC procedure, a clear mapping of Threats and Objectives would be necessary and a description of the means of assurance required. However, this would go beyond the scope of this paper and hence an overview of the Objectives that would be needed to achieve protection from the Threats for each of the Threats listed above is given.

T.ClientForgery/T.BallotForgery would be easy to counteract by using a signed client environment, such as a digitally signed Java applet, which can be verified by the browser. If the ballot is created dynamically in the applet, which would be the case in most election environments with different constituencies, the messages from the election server containing the ballot data (eg as XML structures) need to be signed as well, with the public key being hard-coded in the applet. The hard-coded public key would then be subject to applet verification.

T.AuditForgery involves a large number of secure logging mechanisms. The best way would probably be to organise it in a Blockchain-style way.

T.AuthentDataForgery covers two distinct topics: (i) the ID and authentication of the voter when registering as a voter. This would best be achieved with digital signatures and an eID, which offers a high level of assurance in itself; and (ii) the authenticity of a token used in a Token protocol, which would be achieved by applying blind digital signatures on the token including one by an independent election observer (cf. [8] and [13]) as well as a secure symmetric encryption of the voting token, when stored locally.[5]

T.OwnVoteMisrepresent protection shall ensure that the vote is presented to the voter requesting individual verifiability as it actually is stored in the ballot box. This requires a cryptographic concatenation between the vote and the authentication data, whereby the latter depends on the protocol family chosen.

T.VoteModify protection shall enable to detect the "swap" of a vote that was submitted on a set of authenticated data. Furthermore, this is best prevented by a cryptographic concatenation between authentication data and vote.

T.VoteMultiple firstly requires a strict voter authentication process and the concatenation of whatever means of authentication needed for the vote, depending on the protocol. Should multiple replacement votes be possible, the old votes must clearly and irrevocably be invalidated and the invalidation flag be part of the concatenation information.

T.PhantomVoters requires a voter roll that is cryptographically secured, not just by way of data encryption, but also by a Blockchain-like concatenation of voters to immediately detect fraudulently, or erroneously, inserted voters.

T.InsertPhantomVotes requires a strong cryptographic link between authentication data and the vote to detect such votes in a counting process that follows the guidelines outlined in Section 4 with T.Miscount.

T.UnlawfulResultAccess shall be prevented by encrypting the votes with the public key of an asymmetric key pair when the vote is cast and by splitting the private key part needed to open the ballot among the election committee members.[6]

T.Miscount shall require for the original counting process and T.NoIndependentRecount shall require for the recount independent of the eVoting system itself:

(i) A strong link between authentication data and the vote implemented by a cryptographic concatenation;
(ii) Verification that the vote has not been manipulated;
(iii) Verification that the votes have not been added or multiple votes cast by the same voter;
(iv) In both cases the votes of course also have to be counted.

5 Such as AES in FIPS 197, cf. [17]
6 For an overview, cf. [16]

Arguably, the best way to ensure an uncorrupted vote count is to combine general verification of the result and individual verification of the result. In many cases, this clearly implies violation of voter secrecy. However, Token protocols may provide this ability without compromising voter secrecy, thereby strengthening general verification by individual verifiability of the end result.[7] In such cases, any forger would always run the risk of being discovered by some voters individually checking whether their vote correctly entered the tally.

T.LinkVoterVote protection shall prevent the disclosure as to which voter a vote belongs. In this context, it is immaterial, when the breach occurs. It may occur before vote counting has even started or it may occur sometime after the election closed, possibly even based on backup data. The implementation of this requirement shall, on a protocol and a system level, not interfere with general and individual auditability of the vote.

T.TimeManipulation shall be prevented by a set of multiple secure and synchronised time sources for the eVoting system.

6. Conclusion and Further Work

This paper attempts to start a discussion process towards the establishment of a Common Criteria Protection Profile that can be used as a basis for system certification focusing on the two main areas of progress between of CM/Rec(2017)5 compared to Rec(2004)11, namely verification and voter secrecy. The discussion of course needs to be expanded by mapping Objectives and Threats more closely and by including non-Target-of-Evaluation-relevant objectives. Also, the entire area of assurance requirements needs to be discussed. For sheer space constraints, this paper cannot be more than a first step, however it appears to be worthwhile to make the effort in order to achieve more reliable and trusted eVoting solutions.

7. Bibliography

[1] Common Criteria for Information Technology Security Evaluation, v3.1. Release 5, Part I – II and Annexes, download from https://www.commoncriteriaportal.org/cc/

[2] Common Criteria for Information Technology Security Evaluation – List of certified products, download from https://www.commoncriteriaportal.org/products/

[3] Bundesamt für Sicherheit in der Informationstechnik, List of Certified Protection Profiles, download from https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Zertifizierung-und-Anerkennung/Zertifizierung-von-Produkten/Zertifizierung-nach-CC/Schutzprofile-Protection-ProfilesPP/SchutzprofileProtectionProfiles_Aktuell/schutzprofile_pps_aktuell _node.html

[4] Recommendation Rec(2004)11 adopted by the Committee of Ministers of the Council of Europe on 30 September 2004 and explanatory memorandum, http://www.coe.int/t/dgap/goodgovernance/Activities/Key-Texts/Recommendations/Rec(2004)11_Eng_Evoting_and_Expl_Memo_en.pdf

[7] For a cryptographic consideration, cf. [8]

[5] CHIANG, L., Trust and security in the e-voting system, Electronic Government an International Journal, 2009, 6(4):343-360

[6] ANTONIOU, A., KORAKAS, C., MANOLOPOULOS, C., PANAGIOTAKI, A., SOFOTASSIOS, D., SPIRAKIS, P. and STAMATIOU, Y. C., A Trust-Centered Approach for Building E-Voting Systems, International Conference on Electronic Government, EGOV 2007: Electronic Government LNCS, volume 4656 pp 366-377

[7] BAGNATO, D., The Impact of the Council of Europe Recommendation CM/Rec(2017)5 on Evoting Protocols found in A. Nemeslaki, A. Prosser, D. Scuola, T. Szadeczky (eds.), CEE e/Dem and e/Gov Days 2019, Cyber Security and eGovernment, Wien, 2019, p. 59

[8] BAGNATO, D., MÜLLER-TÖRÖK, R. and PROSSER, A., Council of Europe Recommendation CM/Rec(2017)5 and e-Voting Protocol Design, Masaryk University Journal of Law and Technology, 2020-2-6, pp. 275-300.

[9] Recommendation CM/Rec(2017)5 of the Committee of Ministers to member States on standards for e-voting, http://rm.coe.int/0900001680726f6f

[10] Explanatory Memorandum to Recommendation CM/Rec(2017)5 of the Committee of Ministers to member States on standards for e-voting, http://search.coe.int/cm/Pages/result_details.aspx?ObjectID=090000168071bc84

[11] Guidelines on the implementation of the provisions of Recommendation CM/Rec(2017)5 on standards for e-voting, http://search.coe.int/cm/Pages/result_details.aspx?ObjectID=0900001680726c0b

[12] HEIBERG, S., PARSOVS, A. and WILLEMSON, J., Log Analysis of Estonian Internet Voting 2013–2015, International Association for Cryptologic Research, 2015, download from 1211.pdf (iacr.org).

[13] PROSSER. A. and MÜLLER-TÖRÖK, R., E-Democracy: Eine neue Qualität im demokratischen Entscheidungsprozess, in Wirtschaftsinformatik, 44, 2002, 6, p. 545 – 556.

[14] MAATEN, E., Towards remote e-voting: Estonian case in A. PROSSER AND R. KRIMMER (eds), Electronic Voting in Europe – Technology, Law, Politics and Society, GI-Edition, Lecture Notes in Informatics, p. 83-90.

[15] PROSSER, A., Transparency in eVoting - Lessons learnt. Transforming Government: People, Process and Policy 8 (2): 171-184.

[16] PROSSER, A., KOFLER, R., KRIMMER, R. and UNGER, M. K., 2004. Implementation of Quorum-Based Decisions in an Election Committee. Proceedings of DEXA/EGOV in TRAUNMÜLLER, R.(ed.): Lecture Notes in Computer Science LNCS 3183, Springer, Berlin.

[17] https://csrc.nist.gov/publications/detail/fips/197/fina