# Development of a Formal IT Security Model for Remote Electronic Voting Systems

Rüdiger Grimm[1], Melanie Volkamer[2]

[1]Forschungsbereich IT-Risk-Management
Universität Koblenz-Landau
grimm@uni-koblenz.de

[2]Institut für IT-Sicherheit und Sicherheitsrecht
Universität Passau
volkamer@uni-passau.de

**Abstract:** Remote electronic voting systems are more and more used - not so much for parliamentary elections, but nevertheless for elections on lower levels as in associations and at universities. In order to have a basis for the evaluation and certification, in Germany a Common Criteria Protection Profile [PP08] is developed, which defines basic requirements for remote electronic voting systems. This Protection Profile requires a rather low evaluation depth (EAL2+). For elections on higher levels an appropriate adjustment of the evaluation depth is recommended. In its first part this paper points out that increasing the evaluation depth beyond EAL5 is not possible at present, since EAL6 requires formal methods and in particular a formal IT security model. Such a formal model does not exist yet. In the second part, this paper proposes a first step to an IT security model for remote electronic voting systems, which, however, considers only a subset of the security objectives defined in the Protection Profile [PP08].

## 1 Introduction

Over the last two years, the Gesellschaft für Informatik (GI – the German society of computer scientists) has developed a Protection Profile (PP) for a basic set of security requirements for remote electronic voting systems [PP08] in cooperation with the Bundesamt für Sicherheit in der Informationstechnik (BSI – German Federal Office for Information Security) and the German Research Center for Artificial Intelligence (DFKI). The Protection Profile is based on the Common Criteria [CC06]. It defines a minimum set of security objectives, which every remote electronic voting system has to ensure and a set of assumptions to the environment, in which the system is used. A remote electronic voting system certified against this Protection Profile [PP08] assures a secret, free, equal and universal election only under the condition that the system is used in an environment where the defined assumptions hold.

The Common Criteria (CC) together with the Common Evaluation Methodology [CEM06] define how the compliance of a particular system with the defined security objectives has to be evaluated. The CC differentiates between different evaluation depths. They distinguish between evaluation assurance level (EAL) 1 to 7+, whereby 7+ means the most intensive evaluation. Generally, the deeper this evaluation goes, the higher is the trustworthiness into the certified system. The scope of the system to be evaluated, the evaluation complexity, and the evaluation methods rise with rising EAL level. The Protection Profile, which defines a basic set of security requirements for remote electronic voting systems, requires the assurance level EAL2+ which is characterised by the following aspects:

- Execution of independent and structured tests by the evaluator
- Analysis of the documentation up to the high-level design and the interface specification
- Analysis of the strength of the functions
- Search for obvious vulnerabilities by the evaluator
- Presence of a configuration system
- Evidence of secure system delivery procedures

EAL2+ is certainly sufficient for elections in associations, schools and universities, but not for elections on higher levels and in particular not for parliamentary elections. Thus, for example, the persons in charge of the Protection Profile, which define requirements for the digital election pen[36] [PP06][37], require EAL3+[38]. Some critics demanded EAL4 and even higher.

[36] The digital election pen had been planned for the citizenry election in Hamburg in February 2008.
[37] The Protection Profile is based on the Common Criteria version 2.3.
[38] The Protection Profile required EAL 3 augmented with the following components: ADV_SPM.1 (Informal TOE security policy model) and AVA_MSU.3 (Analysis and testing for insecure states) - replacing AVA_MSU.1.

In the past, systems have been predominantly evaluated according to evaluation assurance levels equal or below EAL4+, since starting from the EAL5 semi-formal and/or formal methods are required. The application of such methods causes substantial additional effort for manufacturers and evaluators. The decision for such a high evaluation assurance level should be made before starting the development because (semi-)formal methods cannot be implemented in the follow-up (the effort to do so in the follow-up is as large as a complete new development). However, EAL5 provides a substantial increase in the trustworthiness of certified systems compared to EAL4, because a semi-formal description of the system design as well as a more modular and therefore better analysable architecture is demanded. A corresponding increase can be identified from EAL5 to EAL6 because the semi-formal specification languages are replaced by formal specification languages. ”Past experiences show that a formal modelling of the security policies given as a formal security model may lead to an increase of confidence in the security of the product that obeys these security policies.” [DFKI02]

Starting from EAL6, the Common Criteria component ADV_SPM.1 has to be ensured, which demands the use of a formal IT security model. Moreover, the component requires a consistency proof (in form of a mathematical proof) for the model itself and a compliance conformance between the system specification and the defined model. To do so, it is possible to use already published and established formal IT security models[39] as a whole or in parts. If no suitable formal IT security model exists, such a model must be developed.

The latter case holds for remote electronic voting systems. Therefore, such a formal IT security model has to be developed before an evaluation according to EAL6 and/or 7 can be aimed. In the context of this article we point out, by the example of some concrete security objectives defined in the Protection Profile, how such a formal IT security model can be designed.

In the further contribution, the definition of an IT security model is introduced (see chapter 2), then it is discussed whether existing IT security models can be applied (see chapter 3). Subsequently, security objectives from the Protection Profile are identified, which are considered for the definition of a formal IT security model (see chapter 4), and afterwards a formal IT security model is developed and proven to ensure all characteristics of an IT security model (see chapter 5). The paper closes with the proposal of future work activities and a short summery (see chapter 6).

[39] Examples for available and established IT security models are: Bell/LaPadula model, the Clark Wilson model, and the Biba model.

## 2 IT Security Model (General Introduction)

**Model Definition**. According to [Grimm08], IT security models define system states and state transitions, differentiate between secure and insecure states, and explain under which circumstances secure states are reached. An IT security model can be more or less formal. All IT security models contain the following *five description elements*:

1. The definition of a superior security objective
2. The specification of secure system states[40] which represent together the superior security objective
3. A trust model, describing a set of assumptions about the environment in which the system is used and under which the set of secure system states is equivalent to the superior security objective.
4. A set of permitted state transitions
5. A security theorem, claiming that applying any permitted state transitions to any secure state necessarily transfers to a secure state again.

**Explaining the Coherences**. An IT security model has to close the following two gaps:

- between the secure system states and the superior security objective (trust model in 3) and
- between the permitted state transitions and the secure system states (security theorem in 5).

For our purpose the first gap is already closed by the Protection Profile; in particular by

- the security problem definition, including a list of assumptions about the environment,
- the list of security objectives for the system, and
- the discussions in section „security objective rationale“.

Therefore, this aspect is not further discussed in this paper. The second gap is closed by the security theorem with its corresponding proof in sections 4 and 5 of this paper.

[40] The specification of secure system states corresponds to the Common Criteria security objectives (in case of a non formal IT security model).

**Definition of Secure System States and Permitted State Transitions**. The secure states (description element 2) and the permitted state transitions (description element 4) have to be described as accurately and precisely as possible. One informal way to formulate secure states is the definition of security objectives according to the Common Criteria [CC07]. In this case, the security theorem (description element 5) is proven by a linguistically convincing and conclusive argumentation. For applications which require a high security assurance, the definitions of a secure state and of permitted state transitions must be consistent and the corresponding security theorem must hold without any doubt. In this case, it is necessary to specify the secure states and the permitted state transitions in a formal way, and the security theorem must be proven with mathematical means. The formal specification of both together (in description elements 2 and 4) together with the formal proof (in description element 5) represents a *formal IT security model*[41].

In the case of a formal IT security model, a third gap has to be closed - the gap between the linguistically formulated security objectives from the Protection Profile and the formal specification of the secure states. This cannot be formalised, but this is the subject of an argumentative discourse of security and application experts.

**Advantages of the Application of Formal Methods.** The application of formal IT security models has three main advantages:

- No natural language can guarantee an unambiguous interpretation and, therefore, it provides no feasibility to prove consistence in the formulation of secure states and permitted state transitions. Vulnerabilities in the implementation of these are a consequence. In contrast, the application of mathematical established technical equipment, which makes the application of computer-aided proofs possible, enables the definition of unambiguous and inter-subjective secure states and permitted state transitions.

- The development of a formal IT security model is used to identify and remove inconclusive, inconsistent, contradictory, or not enforceable secure states and/or permitted state transitions which cannot be detected with natural language.

- Using natural language for the specification of secure states and permitted state transitions causes similar problems for the evaluator - it is hard and in general not unambiguous to decide whether the implemented security functions are sufficient to ensure the specified secure states and permitted state transitions. Based on a formal specification of the system, it can be formally proven that the specification and later the implementation conform to the formal specification of the secure states and permitted state transitions.

[41] The Common Criteria defines formal security models in the following way: "A formal security model is a precise formal presentation of the important aspects of security and their relationship to the behaviour of the TOE; it identifies the set of rules and practises that regulate how the TSF manages, protects, and otherwise controls the system resources. […] the formal security policy model is merely a formal representation of the set of SFRs being claimed." [CC06]

## 3 Application of Available IT Security Models for elections

To our knowledge, no formal IT security model is available which completely covers the superior security objective of a secure remote electronic election. Caused by the numerous different tasks of a remote electronic voting system, the existence of such a model also seems to be unrealistic. However, the integrity model of Clark Wilson [CW87] and the confidentiality model of Bell-LaPadula [BLP73] can possibly describe partial security objectives.

The Clark Wilson model introduced the separation of duty principle to security modelling. For different partial security objectives in the context of a remote electronic voting system, it might be possible to use the separation of responsibilities in the sense of Clark Wilson. The Protection Profile defining basic security requirements for remote electronic voting systems [PP08] demands, for example:

> ***O.AuthPollworkers****: The TOE implements an authentication function which supports the separation of duty principle for at least two members […]. Thus, at least two poll workers control each other.*

This PP security objective corresponds to the certification rule C3 and the penetration rules E2 and E3, which describe the "internal consistency" of a system in the Clark Wilson model:

- E2: The system has a list mapping users to transaction procedures (user X, TPi, (CDIa, CDIb, CDIc, …)) and ensures that users can only execute transaction procedures according to this list.
- C3: The allocation list from rule E2 complies with the separation of duty principle.
- E3: The system authenticates the user's identity before executing any transaction procedure.

The Bell-LaPadula model prevents confidential information flow to public domains. This is achieved by mandatory access control. This approach could conceivably structure voters, poll workers, ballots and the ballot box in a hierarchical information flow model a là Bell-LaPadula and, thus, to model the secrecy of the vote. These approaches are still open research tasks.

The following chapters will discuss other security objectives defined in the Protection Profile, which cannot be modelled with Bell LaPadula, Clark Wilson or none of the other well-known formal IT security models. Therefore, a new formal IT security model is developed for these PP security objectives. The developed transaction procedures for the penetration of these security objectives could be embedded into a superior separation of duty model according to Clark Wilson. This integration needs to be further analysed in the context of future work.

## 4 Selection of PP Security Objectives

The development of a formal IT security model for remote electronic voting systems is a complex task and happens gradually by adding security objectives, defined in the Protection Profile, step by step. The security model, which will be presented in chapter 5, is a first step accomplished for two selected security objectives from the Protection Profile defining basic security requirements for remote electronic voting [PP08]. This first step illustrates how the further security objectives can be specified formally. The two selected security objectives are:

> **O.UnauthVoter**: *Only eligible voters who have been unambiguously identified and authenticated are allowed to cast a vote that is stored in the e-ballot box.*
>
> **O.OneVoterOneVote**: *It is ensured that (A) each voter can cast only one vote and that (B) no voter loses his voting right without having cast a vote. [….].*

## 5 Formal IT Security Model for Remote Electronic Voting

Different possibilities to model a particular system exist. According to [Grimm08] an IT security model for the above identified security objectives can be described in the following way:

**Definition of the Superior Security Objective (1).** Execution of a secure, equal, universal, direct, secret, and free remote electronic election.

**Definition of a System State.** A system state is represented by a triple of the following three entries:

1. $W$ – Set of eligible voters (those who are listed in the electoral register and have not yet cast a vote).
2. $S$ – Set of (encrypted) votes stored in the e-ballot box.
3. $voter: S \rightarrow M$ – Mapping (encrypted) votes on their electors.
   $M$ is a superset of $W_{total}$, that is, $M \supseteq W_{total}$. $M$ contains any user who tries to access the remote electronic voting system, whether or not this particular user has the right to cast a vote. The function *voter* assigns each (encrypted) vote to its producer (voter).

   *Remark 1*: in the case of postal voting, the function *voter* is realised by the outer envelope which is labelled with the sender's name and address. During the tallying phase, the sender information is checked and is verified whether $voter(s) \in W_{total}$ or $voter(s) \in M \backslash W_{total}$. In the first case, the outer envelope is removed and the inner one containing the vote is put into the ballot box, while in the second case the envelope is destroyed.

*Remark 2*: the values of *voter* are visible only for the last vote (or votes) cast into the e-ballot box, i.e., only for the $s \in S_{i+1} \setminus S_i$. After anonymising S, the values of *voter* cannot be reconstructed. Therefore, in praxis, the *voter* mapping should only be used during state transitions on the $s \in S_{i+1} \setminus S_i$. Secure state transitions are controllable on this "visible subset" $S_{i+1} \setminus S_i$ of $S_{i+1}$ only (see rules for permitted state transitions (4) below). For the "invisible part" $S_i$ of the *voter* mapping on $S_{i+1}$ we define $voter_{i+1}|S_i := voter_i$.

**Initial State.** $< W_{total}, S_0=\{\}, voter_0=\{\} >$ is the initial state.

$W_{total}$ stands for the set of all voters in the electoral register (those who have already cast a vote and those who still have the right to cast a vote). The two empty sets $S_0$ and $voter_0$ stand for the empty e-ballot-box in the beginning and the corresponding empty mapping of the empty box on the users of the voting system.

**Specification of Secure States (2).** It has to be defined which properties represent a secure state. According to chapter 4, the PP security objectives O.UnauthVoter and O.OneVoterOneVote are selected to be specified in terms of formal state properties denoting a secure state:

- **O.UnauthVoter:** $\forall s \in S: voter(s) \in W_{total}$**;** that is, the e-ballot box contains only those e-votes ($s \in S$) from which the corresponding elector ($voter(s) \in Wtotal$) is listed in the electoral register. In order to ensure this, the voter needs to be unambiguously identified and authenticated.
- **O.OneVoterOneVote: (A)** $\forall s, s' \in S: voter(s) = voter(s') \Rightarrow s = s'$**;** that is, whenever the set $S$ of cast votes contains two votes from the same voter, then these two votes are identical. Thus, only one of the stored e-votes is tallied. This means that each voter can cast only one vote.

  **(B)** $\forall x \in W_{total} \setminus W: \exists s \in S: voter(s) = x;$ that is, a voter can only become an elector if his e-vote is stored in the e-ballot box ($s \in S$). Thus, he cannot lose his right to vote without having cast a vote which has been successfully stored in the e-ballot box.

*Remark* It is easy to prove that these three conditions for a secure state are equivalent to the following two conditions: "$W_{total} = W + voter(S)$" (where "+" denotes the disjoint union of sets) *and* "The *voter* mapping is injective." An alternative way to prove the security theorem (5) would be to prove that these two conditions are implied by the permitted state transitions (4). However, we prefer to derive our three conditions of a secure state (2) directly from the following permitted state transitions.

**Trust model (3).** The set of assumptions about the environment and the corresponding reasoning are part of [PP08].

**Permitted State Transitions (4).** A state transition from state $Z_i=<W_i, S_i, voter_i>$ to $Z_{i+1}=<W_{i+1}, S_{i+1}, voter_{i+1}>$ is permitted if one of the following rules holds:

- State transitions in which no vote is cast:
  [rule 1] $W_i = W_{i+1} \wedge S_i = S_{i+1} \wedge voter_i = voter_{i+1}$

- State transitions in which a vote is cast and successfully stored in the e-ballot box, that is, the sets $S$ and $W$ are modified:
  [rule 2] $\exists s \in S_{i+1} : ( voter_{i+1}(s) \in W_i \wedge W_{i+1} = W_i \setminus \{voter_{i+1}(s)\} \wedge S_i = S_{i+1} \setminus \{s\} )$

*Remark 1*: All $m \in M$ can initiate a state transition by casting a vote. However, for not permitted state transitions holds: $m \in M \setminus W_{total}. \Rightarrow W_{i+1} = W_i$ and $S_{i+1}=S_i$.

*Remark 2*: The state transition rules use the *voter* mapping only on its visible part, that is, on $S_{i+1} \setminus S_i$. This makes the transition rules usable in praxis.

**Theorem (5).** For all permitted state transitions starting with the initial state, $Z_0=<W_{total}, \{\}, \{\}>$ holds that any reachable state is a secure state.

**Proof.** The theorem can be proven by mathematical induction. To simplify our notation, we write *voter* instead of $voter_{i+1}$ or $voter_i$, we understand that $voter_{i+1}|S_i := voter_i$. To simplify the main proof, it is helpful to first prove that for all permitted state transitions $Z_0$ to $Z_i$ the following three lemmas L1, L2 and L3 hold. These are now named and proven:

**L1:** $S_i \neq S_{i+1} \vee W_i \neq W_{i+1} \Rightarrow \exists s \in S_{i+1}: (S_{i+1} \setminus S_i = \{s\} \wedge W_i \setminus W_{i+1} = \{voter(s)\})$

***Interpretation:*** *During each permitted state transition according to [rule 2] exactly one new vote is generated and exactly the one associated voter loses his right to vote.*

**Proof for L1:** In the case $S_i \neq S_{i+1} \vee W_i \neq W_{i+1,}$ [rule 2] had to be applied. Therefore, there exists an $s \in S_{i+1}$ for which holds: $S_i = S_{i+1} \setminus \{s\}$: Thus $s$ is the only element in $S_{i+1} \setminus S_i$. Therefore, the first part of the lemma is proven. Moreover, according to [rule 2] the following statement holds for this $s$: $voter(s) \in W_i$ with $W_{i+1}=W_i \setminus \{voter(s)\}$. Thus, $voter(s)$ is the only element in $W_i \setminus W_{i+1}$. Therefore, the second part of the lemma is proven.

**q.e.d. (L1)**

**L2:** $W_{total} = W_0 \supseteq W_1 \supseteq W_2 \supseteq \ldots \supseteq W_i$

***Interpretation:*** *The set of eligible voters can only decrease.*

**Proof for L2:** This lemma is a trivial consequence of [rule 2].

**q.e.d. (L2)**

**L3:** $\forall s \in S_i : \exists j < i : voter(s) \in W_j \backslash W_i$

***Interpretation:*** *For each vote stored in the e-ballot box, there exists a voting right discarded earlier.*

**Proof of L3:** Application of proof by induction over *i*, starting with *i=1:*

*Induction Base*: For *i=1*: Choose *j=0*, then this case is equal to the special case of L1 with $S_1$ and $S_0$.

*Induction Hypothesis*: L3 holds for some $i \geq 0$

*Induction Step*: For *i+1* holds:

$\forall s \in S_{i+1}$ does either hold $s \in S_{i+1} \cap S_i$ or $s \in S_{i+1} \backslash S_i$. In the first case the statement is true according to the induction hypothesis. In the second case, L1 proves the statement.

**q.e.d. (L3)**

**Back to the main Proof:**

- *Induction Base:* All three secure state properties do hold for the initial state $Z_0$ because $S_0$ and $W_{total} \backslash W_0$ are equal to the empty set.
- *Induction Hypothesis*: The secure state property holds for some state $Z_i$, $i \geq 0$.
- *Induction Step*: It needs to be shown that for all possible states $Z_{i+1}$ reachable by permitted state transitions from $Z_i$ holds that a secure state is reached:
    - [rule 1] $W_i = W_{i+1} \wedge S_i = S_{i+1}$; thus $Z_i = Z_{i+1}$. Therefore, applying the induction hypothesis it holds that also $Z_{i+1}$ is a secure state.
    - [rule 2] $\exists s \in S_{i+1}$: $(voter(s) \in W_i \wedge W_{i+1} = W_i \backslash \{voter(s)\} \wedge S_i = S_{i+1} \backslash \{s\})$

We prove each of the three properties of a secure state separately:

**O.UnauthVoter:**

*Induction Hypothesis*: For some $i \geq 0$ holds: $\forall s \in S_i$: $voter(s) \in W_{total}$

*Induction Step*: Then for *i+1* holds:

$\forall s \in S_{i+1} : s \in S_{i+1} \cap S_i \wedge s \in S_{i+1} \backslash S_i$.

- Case $[s \in S_{i+1} \cap S_i]$: this holds because of the induction hypothesis.
- Case $[s \in S_{i+1} \backslash S_i]$: according to L1 holds: $W_i \backslash W_{i+1} = \{voter(s)\} \Rightarrow voter(s) \in W_i$ and according to L2 holds: $W_i \subseteq W_{total}$, hence $voter(s) \in W_{total}$.

**q.e.d. (O.UnauthVoter)**

**O.OneVoterOneVote(A):**

*Induction Hypothesis*: For some $i \geq 0$ holds: $\forall s,s' \in S_i$: $voter(s)=voter(s') \Rightarrow s=s'$

*Induction Step*: Then for $i+1$ holds:

For all $s$ and $s'$ only the following three possibilities exist:

- Case [$s, s' \in S_{i+1} \cap S_i$]: this holds because of the induction hypothesis.
- Case [$s, s' \in S_{i+1} \setminus S_i$]: according to L1 holds: $S_{i+1} \setminus S_i = \{s\} \Rightarrow s=s'$
- Case [$s \in S_{i+1} \setminus S_i, \wedge \; s' \in S_i$]: according to L1 holds: $W_i \setminus W_{i+1} = \{voter(s)\} \Rightarrow voter(s) \in W_i \setminus W_{i+1}$ and according to L3 holds $\exists j < i : voter(s') \in W_j \setminus W_i$
  Thus, $voter(s) \in W_i$ and $voter(s') \notin W_i$. Thus, both values can never be equal. Thus, the statement holds also in this third case.

**q.e.d. (OneVoterOneVote(A))**

**O.OneVoterOneVote(B):**

*Induction Hypothesis*: For some $i \geq 0$ holds: $\forall x \in W_{total} \setminus W_i$: $\exists s \in S_i$: $voter(s) = x$

*Induction Step*: Then for $i+1$ holds: For $x \in W_{total} \setminus W_{i+1}$, $x$ must be in one of the following sets:

- Case [$x \in (W_{total} \setminus W_{i+1}) \cap (W_{total} \setminus W_i)$]: this holds because of the induction hypothesis.
- Case [$x \in (W_{total} \setminus W_{i+1}) \setminus (W_{total} \setminus W_i)$]: according to L2 holds: $W_{total} \supseteq W_i \supseteq W_{i+1}$. Thus, $(W_{total} \setminus W_{i+1}) \setminus (W_{total} \setminus W_i) = W_i \setminus W_{i+1}$; thus, $x \in W_i \setminus W_{i+1}$; in addition, it holds: $W_i \neq W_{i+1}$. According to L1 holds $W_i \setminus W_{i+1} = \{voter(s)\}$ for $s \in S_{i+1} \setminus S_i$. Then, deduced from $x \in W_i \setminus W_{i+1}$ it holds: $voter(s)=x$; this completes the proof for $i+1$.

**q.e.d. (OneVoterOneVote(B))**

***All together: q.e.d. (Theorem)***

## 6 Future Work and Summary

Currently, a Protection Profile (PP) defining basic security requirements for remote electronic voting [PP08] is accomplished in Germany. This PP demands the evaluation assurance level EAL2+. The current discussions about the evaluation of electronic voting systems in general illustrate that the critics demand a high EAL level. We agree because political elections are the highest property of a democracy. Therefore, we believe that formal methods are well motivated for voting applications. However, concerning an evaluation according to EAL6 or EAL7 there are still a couple of open questions and research tasks to solve (not only concerning remote electronic voting). It is necessary to further discuss the specification of IT security models for remote electronic voting systems.

This contribution demonstrates with two examples how security objectives, defined by the basic profile PP can be integrated into a formal IT security model. Up to a complete formalisation of all security objectives and their integration in a closed IT security model for remote electronic voting systems, substantial research has to be carried out.

## Acknowledgements

We thank Dieter Hutter for his helpful comments on our formalisation method.

## References

[CC06] Common Criteria for Information Technology Security Evaluation, Version 3.1, 2006.

[CEM06] Common Methodology for Information Technology Security Evaluation, Version 3.1, 2006.

[DFKI02] H. Mantel, W. Stephan, M. Ullmann, and R. Vogt. Leitfaden für die Erstellung und Prüfung formaler Sicherhetismodell im Rahmen von ITSEC und Common Criteria. Version 1.0c http://david.von-oheimb.de/cs/teach/BSI-Leitfaden_1.0c.pdf, 2002

[Grimm08] R. Grimm, IT-Sicherheitsmodelle. Arbeitsberichte aus dem FB Informatik der Universität Koblenz-Landau, Feb 2008, erscheint in WISU

[PP06] M. Volkamer and R. Vogt. Digitales Wahlstift-System. Common Criteria Protection Profile BSI-PP-0031, http://www.bsi.de/zertifiz/zert/reporte/PP0031b.pdf, 2006.

[PP08] M. Volkamer and R. Vogt. Core Requirements for Online Voting Systems. Protection profile, German Research Center for Artificial Intelligence, 2008.

[CW87] D. Clark and D. Wilson. A Comparison of Commercial and Military Security Policies. Proceedings of the 1987 IEEE Symposium on Security and Privacy, Oakland, CA. Computer Society Press of the IEEE, Washington DC, 184-194, 1987.

[BLP73] D. E. Bell and L. J. LaPadula. Secure Computer Systems: Mathematical Foundations, and A mathematical model. ESD-TR-73-278, MTR-2547, Vols 1&2. The MITRE Corporation, Bedford, MA, Nov 1973.