Protection Profile for a Voting System Vote Counter

VSVC-PP

Version Draft

Table of content

64

List of Tables

82

List of Figures

92

1 PP introduction

1.1 Introduction

Based on the advice of the commission Electronic Voting at Polling Stations dedicated Protection Profiles have been developed for two devices that can be used to support the voting process. Namely these devices are the ballot printer and the vote counter. They can be used by the voter to make their choice and print it on a ballot paper and to efficiently count the votes.

The current document represents the Protection Profile for the vote counter.

In order to provide a global overview of the process, the current document contains information on

- The procedural view to voting and counting
- The life-cycle of the vote counter
- Assets to be protected by the vote counter
- Subjects that are interacting with the vote counter
- Threats against the assets
- Organizational Security Policies to be fulfilled
- Assumptions that can be made about the intended environment

The whole content of the current document has been discussed and documented based on the principles for voting.. These are as follows:

- Transparency
- Verifiability
- Integrity
- Eligibility to vote
- Freedom of vote
- Secrecy of the vote
- Equal suffrage
- Accessibility

119

120 1.2 PP Reference

Title:	Protection Profile for a Voting System Vote Counter
Contact:	
Version:	Draft
Authors:	
Registration:	
Certification-ID:	
Evaluation Assurance Level:	The assurance level for this PP is EAL 4 augmented.
CC-Version:	
Keywords:	Voting System, Vote Counter

121 1.3 Specific terms

122 The following specific terms are used in the context of this document

Term	Description
Voter	In the context of this document, the voter is regarded as a person that is legitimated to participate in an election.
Choice	The choice of the voter is the primary asset of the ballot printer. The choice means, on the one hand the selection of a party and a candidate, or the answers to the question for a referendum, or a blank choice on the ballot printer (see figure 1).
Vote	From the moment the ballot paper is in the ballot box, in the context of this document it is regarded and described as vote. The vote is the primary asset of the vote counter.
Mode	Modes are dedicated life-phases where the TOE requires or offers interaction.

123 **Table 1: Specific terms**

124

125

126 1.4 TOE Overview

127 1.4.1 Introduction

128 The TOE defined in this Protection Profile is the vote counter that can be used within an election
129 process. In the following chapters, the overall election process that is supported by the vote counter is
130 described.

131 1.4.2 Procedural Overview

132 A simplified overview shows the process as follows: The voter comes to the polling station and
133 legitimates himself as a legitimate voter against the members of the electoral committee, e.g. by
134 presenting their voter card and their identity document. The members of the electoral committee admit
135 the voter to vote. For that the voter is given the possibility to make a vote choice with the ballot printer
136 and print that choice. After the voter has checked whether their choice has been printed correctly on
137 the ballot paper (every print on the paper shall be only plain text that is readable by everyone) the
138 voter puts their choice into a classical ballot box. In the moment where the choice is put into the ballot
139 box, it becomes a vote.

140 Once the voting has ended the count starts. In this phase the electoral committee performs several
141 actions including the counting of the votes deposited in the ballot box. Before opening the ballot box
142 the electoral committee shuts down the ballot printer so that this device cannot be used anymore in the
143 polling station. The ballot papers can then be counted with the vote counter. The vote counter prints
144 the result of the counting and this printout is then attached to the official report. The count phase ends
145 with drawing up an official report by the electoral committee.

146 It is important to understand that the procedure as it is described in this document differentiates
147 between the voter's choice and the vote. The choice in this context means, on the one hand the
148 selection of a party and a candidate on the ballot printer or the selection of an answer to a referendum
149 question or the selection for a blank vote, but also the printout itself until it is put into the ballot box
150 (see figure 1). Thus, this document always refers to the term "choice" to describe the voters' activities
151 until they put their ballot paper into the ballot box. From the moment the ballot paper has been put into
152 in the ballot box, in the context of this document it is regarded and described as a vote.

153 The paragraphs below provide a more detailed overview of the ballot printer and vote counter as well
154 as of the voting process at all.

155 The following figure summarizes the cooperation of the components from a high level perspective.

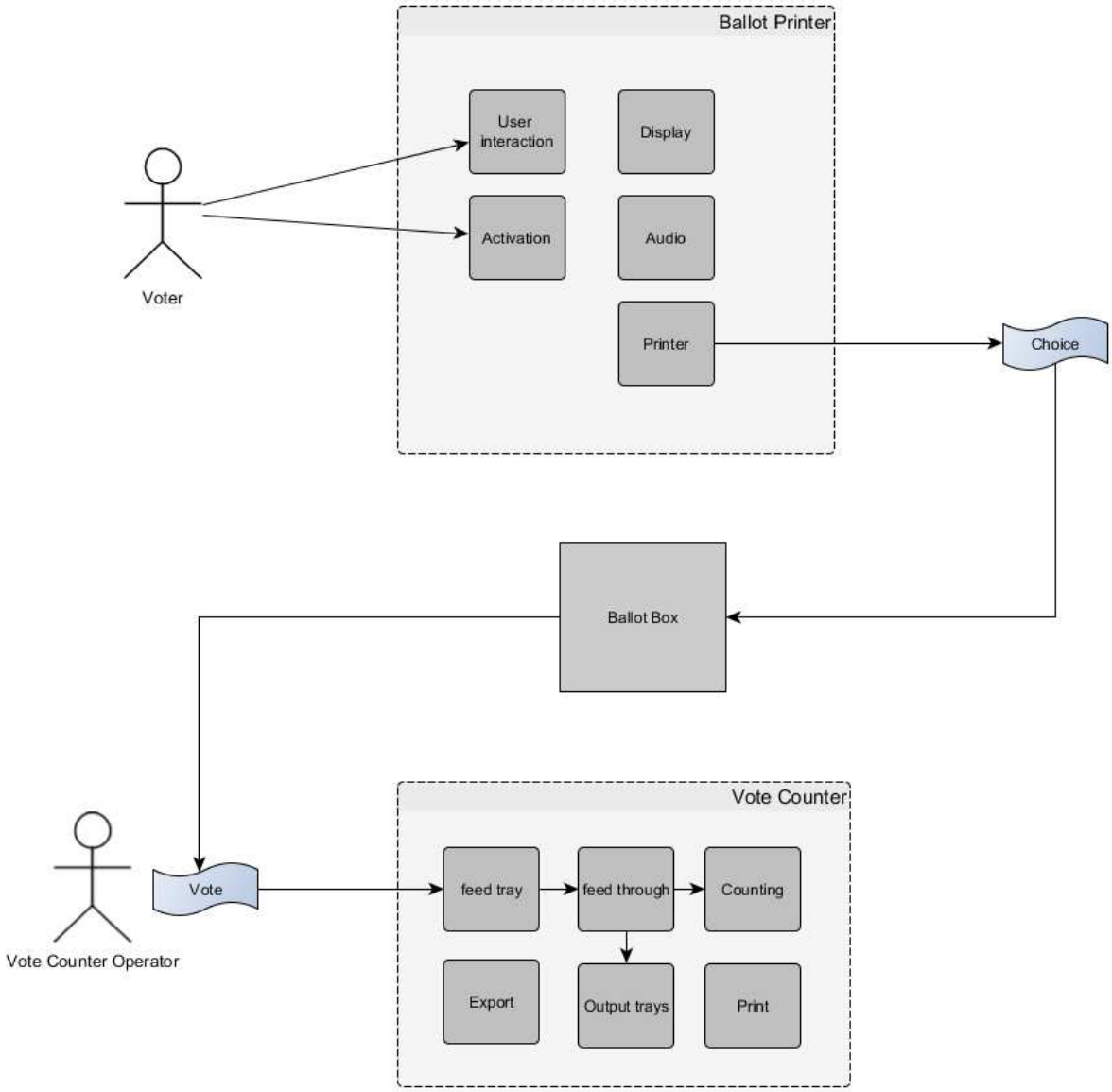


Figure 1: System overview

1.4.3 Detailed overview

From a procedural perspective, it can be distinguished between the phases of voting and counting that are described further within the following chapters.

1.4.3.1 Set up

Before the voting begins the ballot printer needs to be set up (see figure 2).This comprises the placing in the polling station and the connection to electricity. A member of the electoral committee starts-up the ballot printer. He/she shall legitimate and process the start-up by a digital token. The ballot printer requires a self-test and the printout of one or more choices to see whether the ballot printer works correctly. If the electoral committee decides that the ballot printer works correctly the ballot printer is ready for use. The following diagram depicts the set up process of the ballot printer.

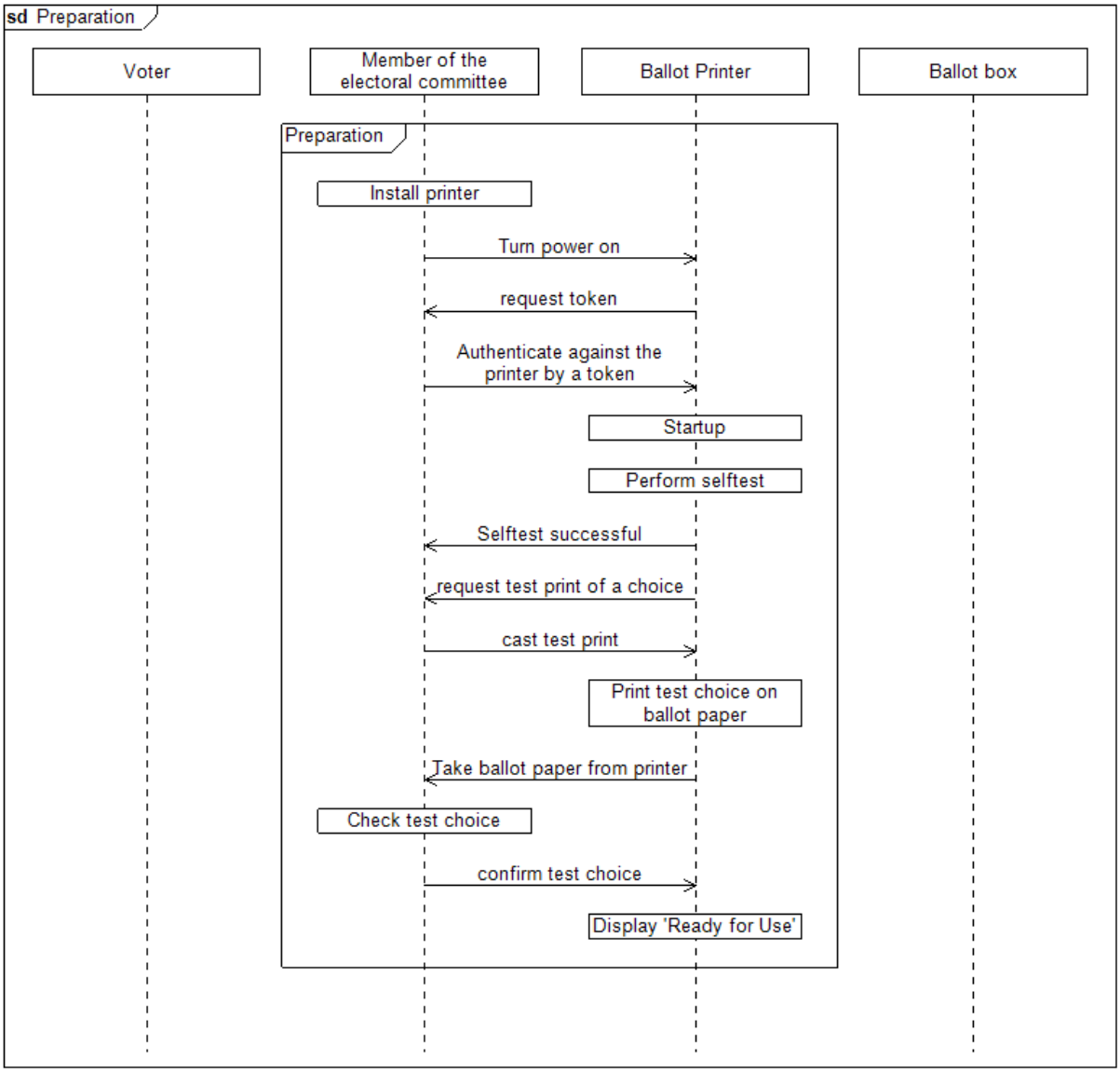


Figure 2: Start process for the ballot printer

1.4.3.2 Voting

On the day of the election, the polling station opens at the time that is defined by electoral laws. A voter that wants to vote, hands over their voter pass and shows a valid ID-document to the electoral committee. In case of multiple elections on the same day, the voter hands over a voter pass for each election the voter is entitled to vote for. The electoral committee checks the voter pass(es), checks if the voter pass is not on the list of invalid voter passes and checks based on the ID-document if the person that wants to vote is the rightful holder of the voter pass(es). If all these checks are successfully completed, the electoral committee gives the voter one or more tokens to activate the ballot printer to make a vote choice for the election(s) the voter is entitled to. The voter receives a token for each election the voter is entitled to cast a vote for. A voter can, in addition to his own vote, cast one or two proxy votes. The proxy votes may only be cast when the voter casts his own vote. For a proxy vote the voter must hand over the voter pass of the proxy giver. On the voter pass the proxy part must have been filled in completely and both proxy giver and proxy receiver must have signed the voter pass. The proxy receiver must also present a copy of an ID-document of the proxy giver.

The voter shall present a token to the ballot printer. The ballot printer swallows the token so that the token can only be used once each time it is handed over to a voter by the electoral committee. The token will activate the ballot printer for the election the voter can make a choice for and guide the

voter through the steps.

Once the voter has made his/hers choice, they will be asked to confirm their choice. In case that the voter confirms their choice, the printer prints the choice on a ballot paper. In the case that the voter does not confirm the displayed choice, the voter can go back in the selection process. After the choice of the voter has been printed the choice made by the voter is deleted from memory.

The voter withdraws the printed ballot paper from the ballot printer and puts the ballot paper in the ballot box. The following diagram depicts the voting procedure.

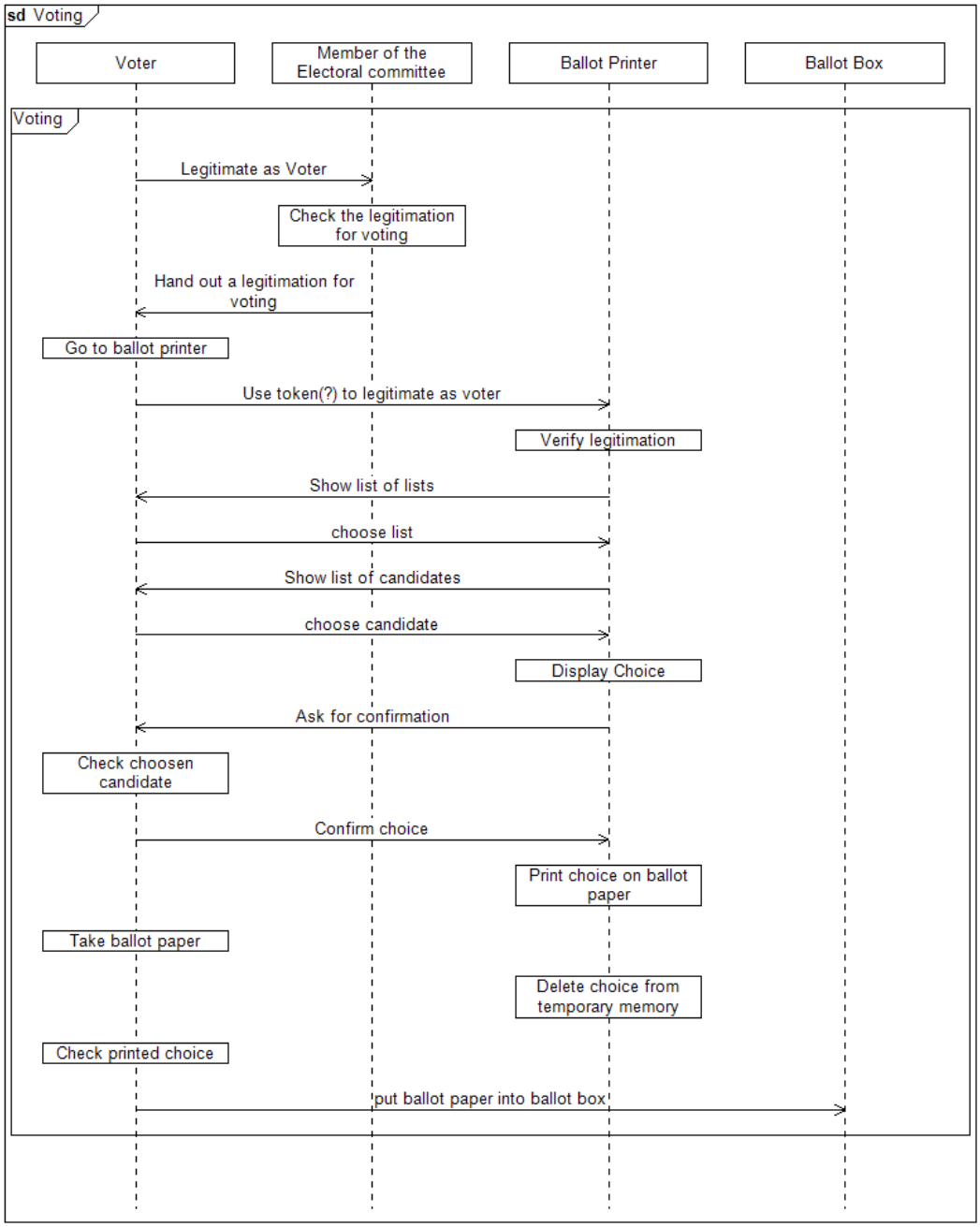


Figure 3: Voting process

Tokens can be re-used by the electoral committee for next voters. To re-use tokens the electoral committee can collect the tokens that have been swallowed by the ballot printer.

To terminate the voting process, the electoral committee shuts down the ballot printer.

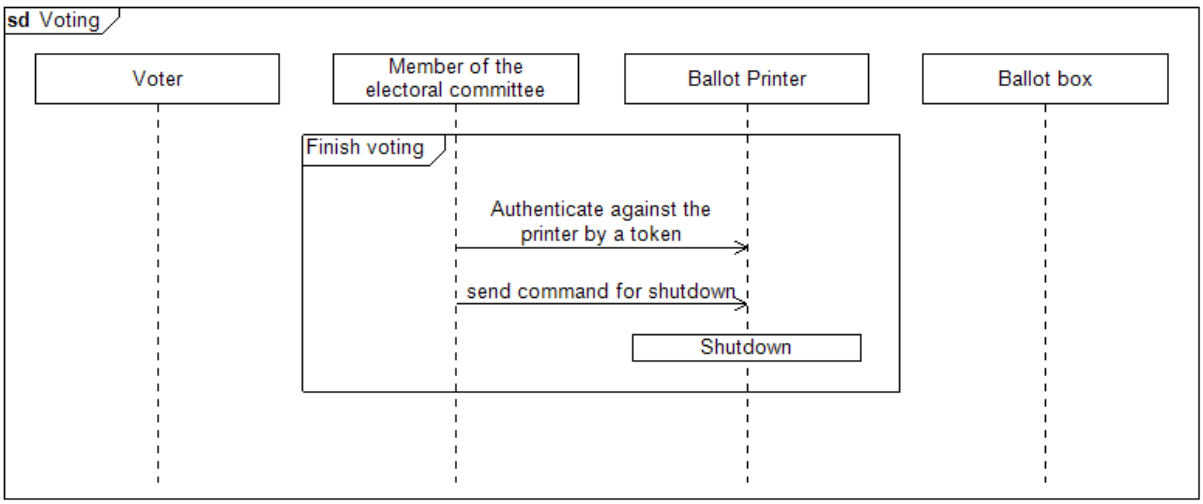


Figure 4: Shut-down of the ballot printer

1.4.3.3 Counting

The voter counter shall be started by a token. The vote counter shall request the number of the polling station or of the ballot box to be entered or a previously set number to be confirmed before it performs its self-test and enters the mode that allows the beginning of the counting process.

During the counting process, for each scanned ballot paper where the vote is recognized the vote counter shall print a consecutive number on the ballot paper. Furthermore it shall save the recognized vote and printed number of every single ballot paper to its log file. When the ballot papers of a ballot box have been put through the vote counter the person that is allowed to operate the vote counter shall confirm this. The counter generates a result of the ballot papers that have been counted and a result (the number of) of the ballot papers that have been rejected because they could not be counted. The results can be printed on paper and can be stored on a digital token. The electoral committee will judge the ballot papers that have been rejected by the vote counter. In the case that the vote counter was able to recognize the vote on the ballot paper and that the consecutive number has been printed, the vote counter shall put this paper in an output tray for successfully counted ballot papers. It shall not put successfully counted ballot papers into a tray for ballot papers that caused problems during the scanning process. The other way round, the counter shall put votes that could not be counted into a tray for those papers and shall not put them into a tray for successfully counted votes. The following diagram depicts the process of counting.

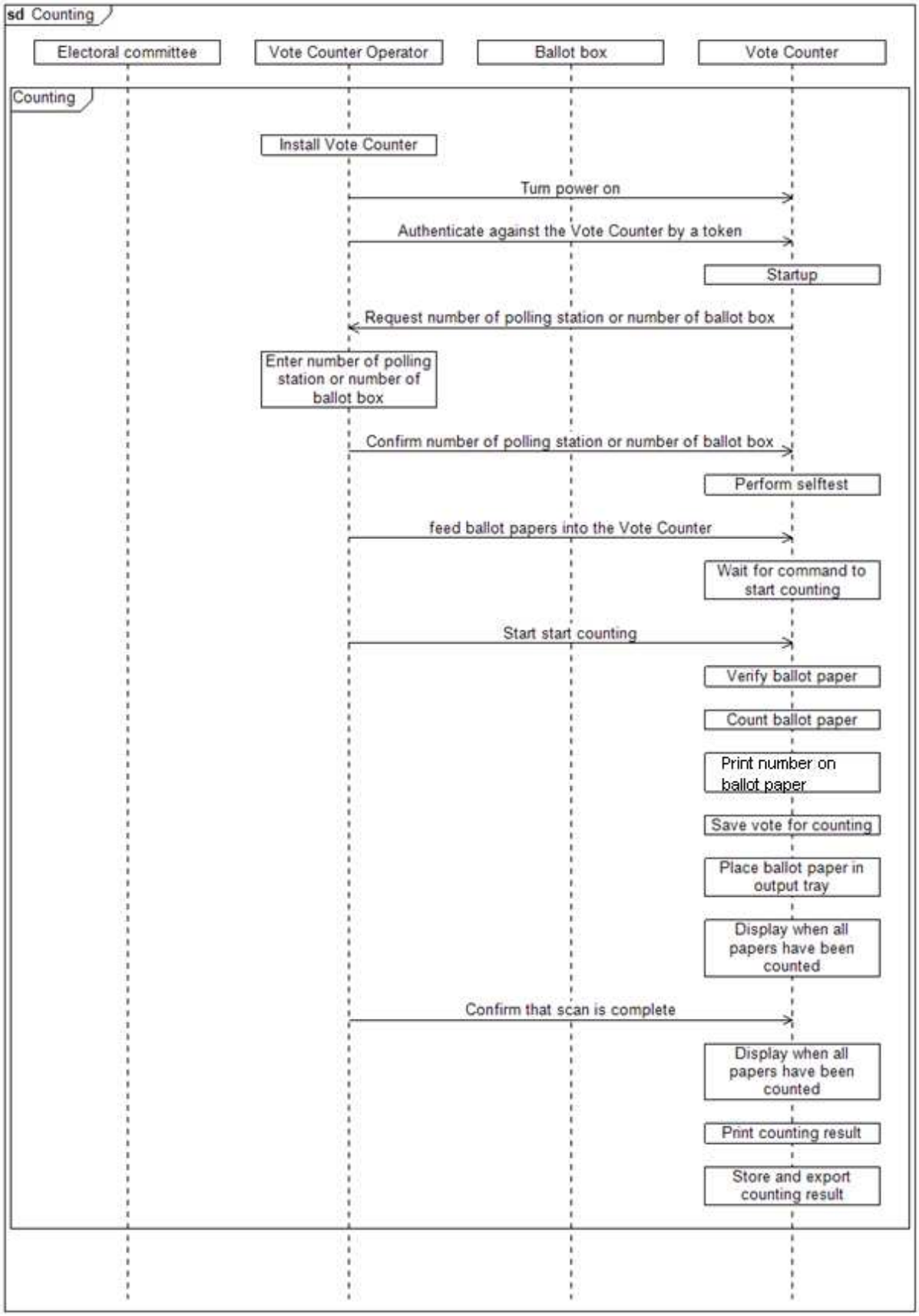


Figure 5: Counting the votes

1.4.4 TOE type

The TOE described in this PP is a counter (vote counter) that is used to count ballot papers within an election process.

1.4.5 TOE physical scope

The physical scope of the TOE comprises the hard- and software that is relevant for the functionality:

- **Casing of the Device:** The casing of the vote counter needs a mechanism to protect the device from intrusion. The ballot vote counter may consist of more than one part. In that case each part shall have its own casing that protects it from intrusion[1].
- **Interface(s) for token:** The TOE provides one or more interfaces that are used for token based role holder authentication.
- **Interface(s) for data transfer:** The TOE provides one or more interfaces that are used for data import and export (election data, token data, configuration data, log-file, counting results).
- **Interface(s) for user-interaction:** The TOE presents activated users the set of interactions they are allowed to perform and guides the user through the process.
- **Scanner**: The TOE includes a scanning unit that is able to scan the ballot paper.
- **Processing unit:** This unit is responsible for processing the counting and all related processes.
- **Output tray(s):** The TOE provides Output tray(s) that allows the sorting of scanned ballot papers.
- **Printing part:** The TOE provides a printing unit that is able to print control lines on the ballot papers and to print the counting result.
- **Security Module:** The TOE includes a security module that shall be used as a cryptographic service provider (it provides key generation, key destruction if required and signature generation)[2]

[1] Some of the requirements in this Protection Profile are dedicated to the case that the TOE may comprise more than one physical part/unit.

[2] The functionality of hashing and signature verification is however provided by the TOE itself.

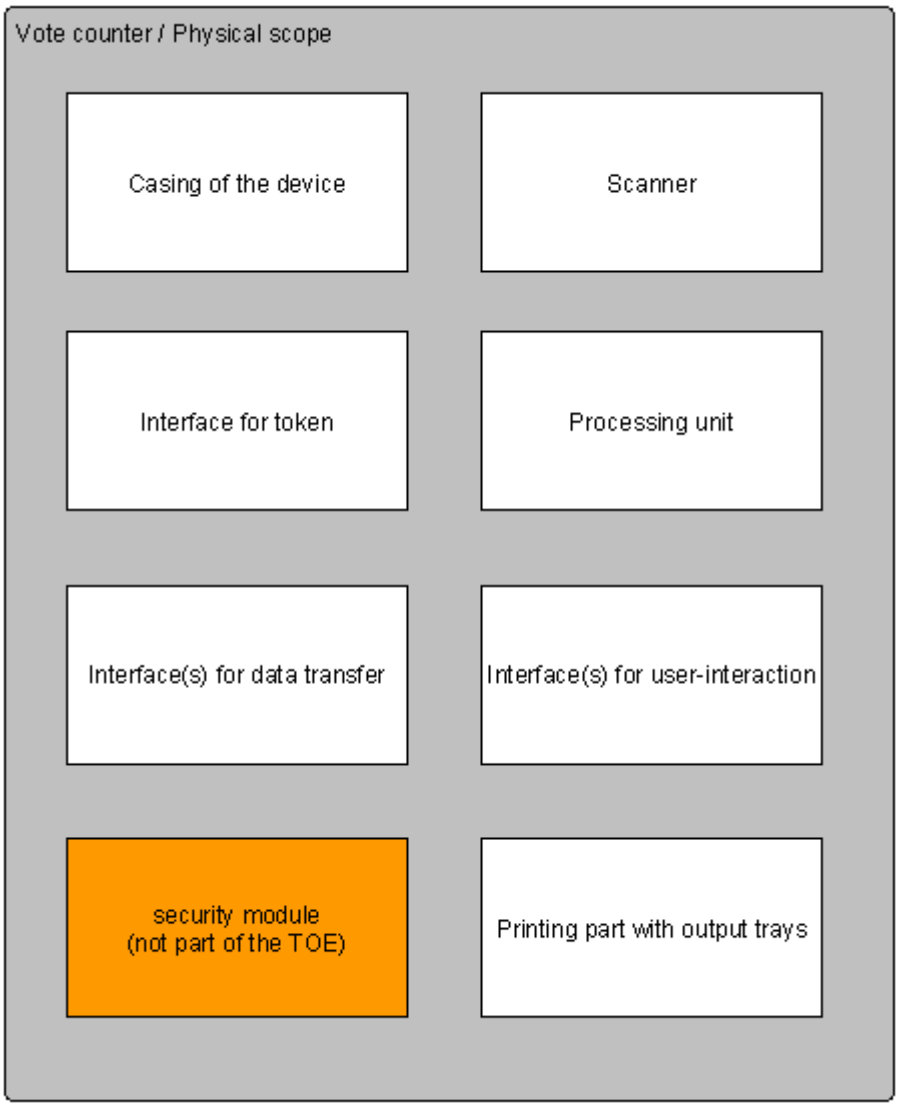


Figure 6: TOE physical scope

Although built into the TOE, the security module itself shall not be part of the TOE. For security modules standard Protection Profiles exist and CC practise is to re-use these and extend them with the additional features and the evaluation level needed. This means that a security module is built in the casing of the TOE and is internally connected to the TOE, but has to be evaluated separately and not in the context of the evaluation of the vote counter. This kind of illustration has been chosen to point out that the security module shall be an internal component that is placed within the casing of the TOE. The security module shall be evaluated according to [PP_SM].

1.4.6 TOE logical scope

The logical scope of this TOE can be defined by its security functions:

- **Token authentication** and **activation:** The TOE is able to authenticate presented token, match token to a defined role and activate dedicated role functionality.
- **Protection** of **integrity** and **authenticity:** Within the whole process, the TOE is able to protect data in terms of integrity and authenticity.
- **Cryptography** that allows the **verification of signatures** on data to be imported by the TOE and **signing of data**, that can be exported**.**

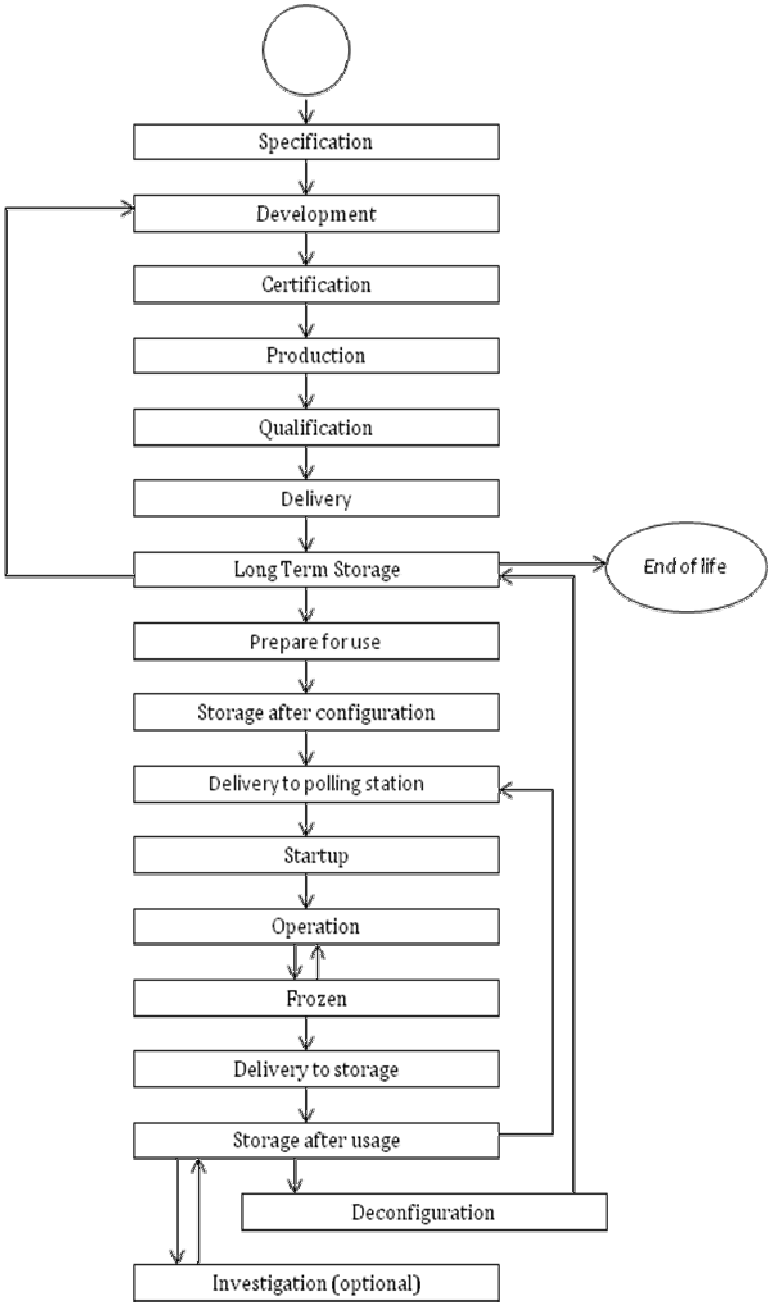
- **Management:** The TOE provides the functionality to manage on the one hand the data that is used for the operation of the TOE (election data) and on the other hand security related data (log-file access, configuration, token management, counting result).
- **Auditing:** The TOE audits and stores defined events and provide the functionality to export the audit logs and to delete them.
- **Self-Protection:** The TOE shall be able to detect whether its hard- or software has been manipulated. In the case that the self-protection mechanism detects an intruder, it shall notify users and switch to a secure state.
- **Self-Test:** The TOE is able to perform a self-test to check, whether the TOE works as specified and allow authorized users to verify the integrity of data, software and hardware.

The TOE uses cryptography that allows the verification of signatures to verify imported data and signing of data to secure exported data. The signing of data is provided by a security module, hence it is not a part of the logical scope of the TOE. See paragraph 1.4.5.

279

280 1.4.7 TOE Life-cycle

281 The following figure shows the life cycle phases for the vote counter.



282
283 **Figure 7: Life cycle for the vote counter**

284

Life Cycle Phase	Description
Specification	During the specification-phase, the public authority that is responsible specifies the requirements that the vote counter shall fulfill. This includes the development of the Protection Profiles for the vote counter.
Development	Based on the specification, the **manufacturer** is responsible for the development of the vote counter in a way that it matches the

Life Cycle Phase	Description
	requirements of the specification. Thus, this phase begins when a **manufacturer** is awarded the contract for the development and ends when TOE samples have been successfully released. Additionally, the vote counter returns from other phases back into the development, when the specification has changed and the **manufacturer** needs to update the devices.
Certification	This phase comprises the evaluation of the TOE samples by an **evaluation body for Common Criteria** and the certificated by a **certification authority**.
Production	After the certification of the TOE samples, the production of the vote counter starts. The **manufacturer** shall ensure that compared to the TOE samples no component of the vote counter is changed in any way whatsoever during the whole process of production.
Qualification	The qualification of every produced vote counter by an independent **evaluator** ensures that the produced vote counters are consistent with the evaluated and certified TOE samples.
Delivery	Once the devices have been qualified, an **Authority for distribution** distributes the devices to the municipalities.
Long Term Storage	After their distribution to the municipalities or after an election, vote counters require a secure long time storage at the **Municipal authority** to ensure that they cannot be manipulated.
Prepare for use	The preparation of the vote counter comprises the configuration of election options (e.g. parties and candidates) as well as a test of the devices whether all components work correct. The configuration shall be done by the **(de)Configurator**.
Storage after configuration	After configuration, the **Municipal authority** will store the systems in a secured area that the municipal authority has designated for this purpose.
Delivery to polling station	The **Municipal authority** will transport the systems to the polling station.
Startup init	The startup of the vote counter on the day of the election is done by the **vote counter operator**.
Operation	In its operational phase, the vote counter is started by the vote counter operator and used to count the votes. Once the counting has ended, the **vote count operator** shuts the vote counter down. If during operation a vote counter's self-protection mechanism registers a manipulation or defect then the vote counter will go the "Frozen"state, both to prevent the vote counter from being used for counting ballots and to protect the information contained therein.
Frozen	After the election, configuration data, logs and counting results shall remain in the vote counter until the result of the election is confirmed by the **Central electoral committee** or in case a criminal investigation has been initiated, after that investigation has been completed.
Delivery to storage	The **Municipal authority** will transport the systems from the polling station to a secured storage location(s) that it has designated for this purpose.
Storage after usage	After the voting, the **Municipal authority** will store the systems in a

Life Cycle Phase	Description
	secured area that it has designated for this purpose. If the central electoral committee decides that a new vote is necessary, the municipal authority will transport the systems back to the polling station again.
Investigation (optional)	In case of malfunction, manipulation or suspicion of malfunction or manipulation, the vote counter needs to be investigated. This investigation will be done by **an authority for investigation**.
Deconfiguration	After the central electoral committee has confirmed the outcome of the election the **(de)Configurator** deletes the counting data, election data and logs from the devices. The devices are then transferred to long-term storage.
End of life	In this phase, the **Manufacturer** destroys the vote counter in a way, that it cannot be used again and that all data is deleted in a secure way.

Table 2: Life-cycle phases and their description

1.4.8 TOE Modes

The life cycle phases can be grouped into dedicated operational modes according to their required functionality. This allows the available functions of the modes to be reduced to the required minimums and reduces the likelyhood of security violations. Furthermore the limitation to a predefined sequence of modes helps to satisfy the security requirements that are implemented in the vote counter. For the vote counter the following operational modes have been defined:

- Election
- Management

The relation between the life-cycle phases and the modes is shown in Table 3:

TOE mode	TOE life-cycle phase
Election	“Operation”,
Management	“Delivery”, “Long Term Storage”, “Prepare for user”, “Storage after configuration”, “Delivery to polling station”, “Startup”, “Frozen”, “Delivery to storage”, “Storage after usage”, “Investigation” and “Deconfiguration”

Table 3: Relation between TOE modes and life-cycle phases

The possible sequence of modes are depicted in Figure 8. In order to activate the “Election” mode it is necessary to present a token that is assigned to a role that is allowed to change the mode of the TOE.

Note: The TOE mode “Election” is not persistent, i.e. will change to “Management” in case of a shutdown of the system or power supply failures.

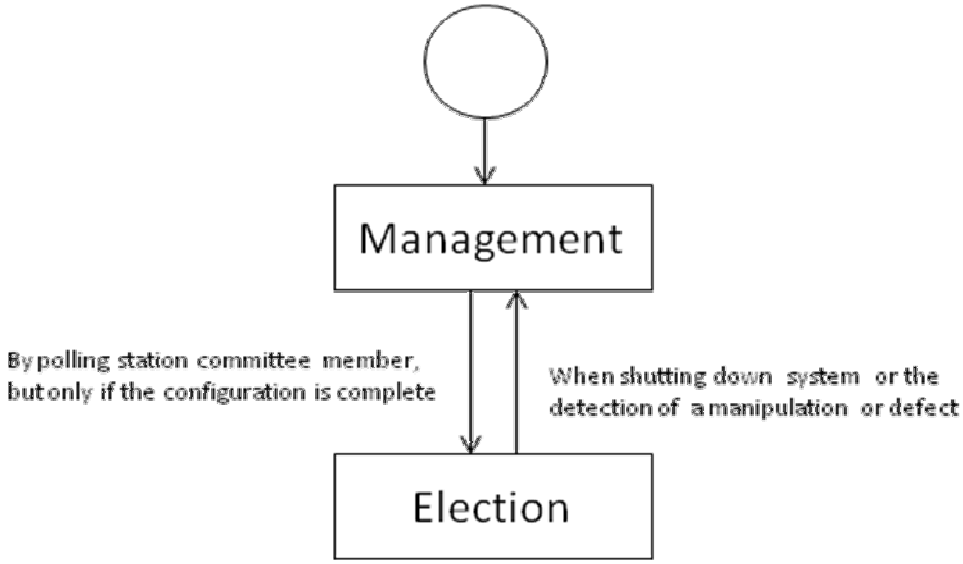


Figure 8: TOE mode diagram of the vote counter

Every mode has the following two authentication sub states:

- NOT AUTHENTICATED: TOE has been powered on, no token present.
- AUTHENTICATED: TOE has been powered on, role holder token authentication has been performed successfully.

The TOE is not aware of the following life-cycle phases:

- Specification
- Development
- Certification
- Production
- Qualification

Application Note:

The TOE starts to exist after production and qualification. During qualification all TOE modes are available and tested. Table 3 shows the relation between the defined TOE life-cycle phases and TOE operational modes.

1.4.9 Authentication Token

The token to activate the modes and to gain access to the vote counter for administrative tasks is not part of the TOE. For such authentication tokens standard Protection Profiles exist and CC practise is to re-use these and extend them with the additional features and the evaluation level needed. The activation tokens shall be based on devices that have been evaluated according [PP-AM].

1.4.10 TOE data structure

The data that is used by the TOE can be divided into two main parts:

- User data
- TOE Security Functionality (TSF) data

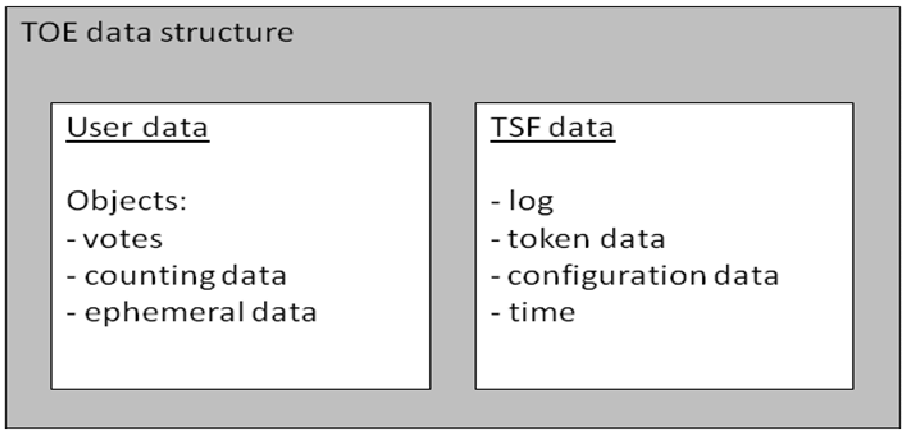


Figure 9: TSF data structure

User data:

User data refers to the data that is processed by the vote counter and that has to be protected in terms of integrity and authenticity. The user data in this context is limited to the votes, the counting results and the ephemeral data associated with it.

It should be noted that the system of authentication of the TOE is based on tokens. Those tokens are treated as users even though the TOE will never get hold of the real user identity.

TSF data:

Refers to all other data that are necessary to operate the TOE. All of the other data does not belong to a dedicated user but is necessary to guarantee the functionality of the TOE, hence is summarised as TSF data. The following list of TSF data summarizes the information that is used in the context of this PP. Note however that this list does not claim to be complete.

- The log file
- Information about the authentication token (i.e. the link between the token ID and the role, public keys)
- Configuration data for election
- The time

2 Conformance Claims

2.1 Conformance statement

This PP requires strict conformance of any PP/ST to this PP.

2.2 CC Conformance Claims

This PP has been developed using Version 3.1 Revision 4 of Common Criteria [CC].

This PP claims conformance to [CC] part 2 extended.

This PP claims conformance to [CC] part 3 extended.

2.3 PP Claim

This PP does not claim conformance to any other PP.

2.4 Conformance claim rationale

Since this PP does not claim conformance to any Protection Profile, this section is not applicable.

2.5 Package Claim

This PP is conforming to assurance package EAL6 as defined in [CC] Part 3 augmented by the use of ALC_DVS.2, AVA_VAN.5 and an explicitly drafted assurance component, ALC_DEL.2.

The SFRs in this PP form a functional package “vote counter functionality” and use SFRs from part 2 of CC plus one extended component named FDP_FPC.1.

3 Security Problem Definition

The Security Problem Definition (SPD) is the part of a PP, which describes

- the **external entities** that are foreseen to interact with the TOE,
- the **assets** which the TOE shall protect,
- the **assumptions** on security relevant properties and behaviour of the TOE's environment,
- **threats** against the assets, which shall be averted by the TOE together with its environment,
- **operational security policies**, which describe overall security requirements defined by the organisation in charge of the overall system including the TOE.

3.1 External entities

The following external entities are allowed to interact with the vote counter in dedicated modes. Those roles have been defined for the use in this Protection Profile.

Role	Description
(de)Configurator	The central electoral committee for an election decides on the admission of lists that can participate in an election and the admission of the candidates that can be put on the lists. The admitted lists and candidates and the admitted question(s) for a referendum are published. The (de)configurator shall check the vote counter before being used during the ballot. The checks that the (de)configurator needs to perform includes (but are not limited to): • Checking the version of the software • Conducting a self-test, including a check of the security of the vote counter • Checking the integrity of the hardware, software and data After these checks have been successfully performed the (de)configurator uploads the list of parties and candidates or the question(s) for a referendum the vote counter requires in the electon mode. The role is also responsible for additional configuration data that is required by the TOE. The (de)configurator then performs a functional test. The vote counter maintains a log file with stored audit events. The (de)configurator is allowed to read and export the information from this log file and other data that is relevant for analysis. Furthermore it falls into the responsibility of the (de)configurator authority to delete the election data after the central electoral committee has announced the outcome of the election or - in case a criminal investigation has been initiated -, after that investigation has been completed.
Vote Counter Operator	The Vote Counter Operator is responsible to start up the vote counter when it is needed for counting. The Vote Counter Operator is allowed to export the counting result.

Table 4: Roles used in the Protection profile

376 ## 3.2 Assets

377

Asset	Description	Need for Protection
Vote	From the moment the ballot paper has been put into the ballot box, in the context of this document it is regarded and described as vote. As soon as the printed choice has been placed into the ballot box, there is no further need to protect the confidentiality of the vote. It must be ensured that a vote is counted and only counted once during one count process (as described in chapter 1.4.2).	• Integrity
Logs	The vote counter maintains a log. Logs must be protected in terms of integrity and authenticity. It is however required that log files in the devices are securely deleted as soon as the results of an election process have been declared or in case a criminal investigation has been initiated, after that investigation has been completed.	• Integrity • Authenticity
Configuration data	The configuration data contains information about the upcoming election or elections (if more than one election takes place on one day) that is going to take place or is taking place that the vote counter has to be used in. It also comprises the list of parties and list of candidates or the referendum question(s) for each referendum that will take place. It shall be protected in terms of authenticity and integrity.	• Integrity • Authenticity
Token data	The TOE is activated by tokens. This means that tokens are presented to the TOE to enable one of the modes described in Table 13 and the corresponding functionality of the role. The TOE shall verify the authenticity of the token, identify the token and the role that is associated with this token and whether this role is allowed in the current mode. In this context, token data explicitly refers to data that is stored in the vote counter. It does not refer to any data that is stored on the token. The roles the TOE shall be able to separate are depicted in Table 4.	• Integrity • Authenticity
Hardware	The hardware of the vote counter can be seen as a dedicated asset. The hardware shall be protected in terms of integrity and authenticity in order to allow a secure operation.	• Integrity • Authenticity
Software	The software of the vote counter can be seen as a dedicated asset. The software shall be protected in terms of integrity and authenticity in order to allow a secure operation.	• Integrity • Authenticity

Asset	Description	Need for Protection
Ephemeral vote counter data	The vote counter may need to work with ephemeral data in the course of the counting process. Such ephemeral data includes but is not limited to • The vote on a ballot paper that is currently processed • Intermediate counting results • Log file information before written to persistent storage This ephemeral data need to be protected in terms of integrity and need to be deleted when the results of an election process have been declared or in case a criminal investigation has been initiated, after that investigation has been completed.	• Integrity
Persistent counter data	The vote counter counts the votes read from ballot papers and keeps the information on those read votes in memory/storage. This cumulated data needs to be protected in terms of integrity.	• Integrity
Export data	The vote counter offers a functionality to export the results of the counting process. This export may happen as a printout and in form of an electronic record. The electronic record needs to be protected in terms of integrity and authenticity.	• Integrity • Authenticity

Table 5: Assets

3.3 Assumptions

In general IT-systems, there is often a need to assume that at least a subset of the subjects that are interacting with the system can be assumed to be non-hostile.

For a voting process however, such assumptions will have to be very limited. Specifically, almost everybody who gets in contact with the vote counter for making choices– either as a user or from an organisational perspective – may have a motivation, the resources and also the opportunity to manipulate (or at least attempt to manipulate) the devices. This motivation does not have to aim to actually manipulate the vote counter, but can also aim to only proof that manipulation is possible, so that the confidence in the reliability of the vote counter is reduced or dropped.

It has therefore been the clear scope in the course of the development of this chapter to put only the absolute minimum level of trust into the users of the vote counter.

Assumption	Description
A.Replacement	It is assumed that a sufficient amount of vote counters are available in case a malfunction occurs and a device becomes un-operational and has to be replaced.
A.SecurityFeature	It is assumed that the ballot paper has a security feature that protects against forged ballot paper. This security feature will be checked by the electoral committee when the number of counted ballot papers is larger than the number of admitted voters and should contribute to prevent that a ballot paper

Assumption	Description
	is counted without the feature.
A.Expendable	It is assumed that any expendable material that is used by the vote counter is available at an adequate amount.
A.Environment	It is assumed that the vote counters are operated in a controlled environment. Specifically, it is assumed that access to the area where voting takes place is controlled. During storage, configuration and transportation it is assumed that the ballot printer is save. It is further assumed that before the voting process starts the feature to verify the authenticity of the ballot printer will be used[3]. It is assumed that the electoral committee makes a count of the admitted voters and compares this to the number of votes cast in order to ensure that no additional votes have been counted. Further a number of random ballot papers are checked for proper counting. This way a significant manipulation should be detected.
A.Admin	It is assumed that the administrative roles[4] that interact with the vote counter have been trained with respect to their responsibilities. However it is not be assumed that the vote counter operators are skilled in detection of attempts of attacks on the vote counter or are able to detect that there is a malfunction. Furthermore it is assumed, that storage and distribution of the tokens falls into the responsibility of an administrative role and that therefore, for the vote counter, it can be assumed that only persons that are allowed to have access to the tokens can have that access. Storage and distribution in this case refers on the one hand to the phase when an election is prepared and the tokens are distributed to the administrative roles that operate the TOE.
A.Token	It is assumed that the tokens for administrative purposes are evaluated according to [PP_AM].
A.SM	It is assumed that the TOE has a built-in security module that provides the required cryptographic functionality and has been certified according to [PP_SM].

Table 6: Assumptions

3.4 Threats

The following section identifies the threats that are posed against the assets handled by the TOE. The description contains on the one hand the primary target of the attack as well as the threat agent that might conduct the attack. In this context, the term **general attacker** is used. The general attacker can be characterized as an attacker with high attack potential in terms of Common Criteria. He must not have the aim to actually manipulate the vote counter, but can merely aim to proof that manipulation is possible, so that the confidence in the reliability of the vote counter is reduced or dropped This means that the attacker

- May spend a relevant amount of time in order to prepare/conduct an attack
- Is highly skilled
- Has internal knowledge about the vote counter

[3] The assumptions regarding storage, configuration, transportation and the verification of authenticity are not realistic and enforceable (from a security point of view). These assumptions in the current Protection Profile are necessary because there are no known other physical protection mechanisms to warrant the integrity of the hardware of the vote counter.

[4] This basically refers to everybody interacting with the devices but the voter

405 • Has access to the devices that is almost unlimited (even though the devices may not be in their
406 operational mode)
407 • Has access to sophisticated equipment.
408
409

Threat	Description
T.MultipleVotes	An attacker could try to achieve that the vote of a voter who is in principle allowed to vote is counted multiple times within one counting process. Furthermore, the attacker could try to achieve that votes that have not been printed by a ballot printer are counted. These additional votes do not have to be the same. The attacker in this scenario can either be the voter who is trying to achieve the goal of the attack in the course of the voting process. On the other hand the attack can also be prepared or conducted by the vote counter operator or a general attacker. Also a combination of these attackers is possible.
T.WrongVote	An attacker could try to achieve that the vote of a voter is counted for a wrong candidate. The attacker may utilize functionality of the ballot printer to printout a choice in a way that will cause the vote counter to count wrong. It is further possible that an attacker in this scenario manipulates the ballot paper that has been (correctly) produced by the ballot printer in a way that will cause the vote counter to count wrong before the manipulated ballot paper is inserted into the ballot box. The attacker in this scenario can either be the voter who is trying to achieve the goal of the attack in the course of the voting process for themselves or for subsequent voters. On the other hand the attack can also be prepared or conducted by the vote counter operator who manipulates the ballot papers to achieve this goal or a general attacker. Also a combination of both attackers is possible.
T.WrongPoll	An attacker could try to achieve that the configuration data that the vote counter uses is wrong. This explicitly includes the case that the configuration data of the vote counter is not identical with the configuration data that was used by the ballot printer. This could lead to a situation in which a significant amount of votes are not counted as voters would vote for parties and candidates who are not allowed to participate in the election. Further, this could lead to a malfunction in counting the votes as the vote counter would try to recognize votes for parties and candidates that are actually not allowed to participate in the election. The attacker in this scenario can either be an administrative user who is trying to achieve the goal of the attack in the course of the voting process. On the other hand the attack can be prepared by a general attacker. Also a combination of these attackers is possible.
T.WithholdVote	An attacker could try to achieve that a cast vote is withhold. With other words, a vote of a voter is not counted by the vote counter. The attacker in this scenario can either be the voter who is trying to achieve the goal of the attack in the course of the printing process but for all subsequent voters. On the other hand the attack can also be prepared or conducted by the vote counter operator who manipulates the ballot papers to achieve this goal or a general attacker. Also a combination of these attackers is possible.

Threat	Description
T.ManipulatedCounting	An attacker could try to manipulate the outcome of the counting of the votes by the vote counter. This is primarily the printed version of the outcome that is attached to the official report. But the attacker may also try to manipulate the electronic outcome of the counting or the stored (log) data. This attack is primarily directed against the vote counter. An attacker in this scenario could either be the vote counter operator or a general attacker.
T.Log	An attacker could try to gain access to the log files in order to manipulate, or delete them. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the counting process. On the other hand the attack can also be prepared or conducted by a general attacker. Also a combination of both attackers is possible. As part of this attack, the attacker could try to modify the internal clock.
T.UnauthorizedAdmin	An attacker in this scenario could try to use administrative functions that he is not authorized for. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the counting process. On the other hand the attack can also be prepared or conducted by a general attacker. Also a combination of both attackers is possible.
T.UnauthorisedUse	An attacker in this scenario could try to use the vote counter without authorization. Without any form of authorization an attacker could use the vote counter to find vulnerabilities. The attacker in this case will be a general attacker because the authorized voter counter operator is allowed to use the vote counter.
T.WrongModeChange	An attacker in this scenario could try to manipulate the mode changes the vote counter is allowed to go through. The impact of this attack would be that the attacker has access to functionalities that should not be available at this point of time. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the counting process. On the other hand the attack can also be prepared or conducted by a general attacker. Also a combination of both attackers is possible.
T.IncorrectNumber	An attacker could try to manipulate the vote counter in a way that the print of the consecutive number on the ballot paper does not correspond with the counting result. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the counting process. On the other hand the attack can also be prepared or conducted by a general attacker. Also a combination of both attackers is possible.
T.Hack	An attacker in this scenario interacts with the vote counter , its interfaces or parts of it to find vulnerabilities and even tries to exploit vulnerabilities. This may compromise security and affects all assets. The goal of the attacker may be just to prove that there are vulnerabilities without compromising security or any assets and by doing so bring the whole voting system in discredit. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the counting process. On the other hand the attack can also be prepared or conducted

Threat	Description
	by a general attacker. Also a combination of both attackers is possible.
T.System_Forgery	An attacker in this scenario replaces the vote counter, or parts of it, with counterfeit parts or presents false parts as genuine vote counter parts. This threatens vote counter integrity, but may also result in compromise of assets. The goal of the attacker may be just to prove that a complete vote counter or parts can be replaced by non authentic ones without being noticed and by doing so bring the whole voting system in discredit. The attacker in this scenario can either be the vote counter operator who is trying to achieve the goal of the attack in the course of the voting process. On the other hand the attack can also be prepared or conducted by a general attacker. Also a combination of both attackers is possible.

410 **Table 7: Threats**

411

412 3.5 Organizational Security Policies (OSPs)

413 Organizations security policies (OSPs) are a means to require functionality from a system that is
414 considered in this Protection Profile even though such functionality is not directly needed to mitigate
415 an attack against the system.

416 The following OSPs will have to be implemented by the devices in this system.

417

OSP	Description
OSP.Log	The vote counter shall maintain a log of security relevant events. Those events shall include all actions with have been performed on the vote counter.

418 **Table 8: Organizations security policies**

419

420 4 Security Objectives

421 4.1 Security Objectives for the TOE

Objective	Description
O.Process	The TOE shall ensure that it generates a reliable and correct result of the number of counted votes on the ballot papers.
O.Integrity	The TOE shall ensure the integrity of the counted data as long as it is remains inside the TOE. This refers to the ephemeral data that is used to generate the counting result as well as to the data that is permanently stored in the TOE (such as the log file).
O.Log	The TOE shall generate audit events for each action that is performed by the TOE. The integrity of the audit log file shall be ensured and be accessible for specific roles in dedicated modes.
O.Management	The TOE shall provide functions to authorized roles within dedicated modes to manage the configuration of the TOE or to use/manage security features. • Authorized roles shall be able to upload the election data (parties and candidates or referendum question(s)) • Authorized roles shall be able to upload the token data that is responsible for the access control. • Authorized roles shall be able to delete the election data and log files after the central electoral committee has confirmed the outcome of the election or in case a criminal investigation has been initiated, after that investigation has been completed. • Authorized roles shall be able to read the audit logs.
O.DataExchange	The TOE shall provide an interface that allows administrative roles export and import of signed data. • The TOE shall be able to verify imported election and token data in terms of authenticity and integrity and only accept this data after verification • The TOE shall be able to verify software/firmware updates in terms of authenticity and integrity and only accept this data after verification • The TOE shall be able to sign the log file to ensure its authenticity and integrity after the export. • The TOE shall be able to sign its exported counting result to ensure its authenticity and integrity after the export.
O.Selfprotection	The TOE shall implement functions to protect itself against manipulation, forgery and malfunction. The vote counter shall have features to detect physical tampering and verify its authenticity. This functionality shall specifically protect against modification of hardware, software, the use of test modes or of existing back doors even if this does not affect security or assets. Furthermore, the manipulation of the power supply shall not lead to a successful attack.
O.AccessControl	The TOE shall control access to the TOE and to its functionality based on roles and dedicated modes as described in chapter 1.4.8. This means that the TOE has predefined mode changes and within each mode only dedicated roles are allowed to interact with the TOE.

Objective	Description
	The TOE shall authenticate a digital token that are associated with a dedicated role (This does not mean that the TOE gains any information about the user of the TOE) and check whether this token can activate the TOE in its current mode. The TOE shall ensure that it can only be activated if the role that is represented by the token is authorized for this mode. As part of the login process of roles the TOE shall – before login – present a banner message on the authorized use of the TOE and – after successful login of an administrative role – information about the last logins of that role.

422 4.2 Security objectives for the operational environment

Objective for environment	Description
OE.Replacement	It shall be ensured that a sufficient amount of vote counters are available in case a malfunction occurs and a device becomes un-operational and has to be replaced.
OE.SecurityFeature	A ballot paper that is printed by a ballot printer contains a security feature that protects against forged ballot paper. This security feature will be checked by the electoral committee when the number of counted ballot papers is larger than the number of admitted voters and should contribute to prevent that a ballot paper is counted without the feature.
OE.Expendable	It shall be ensured that any expendable material that is used by the vote counter is available at an adequate amount.
OE.Environment	It shall be ensured that the vote counter is operated in a controlled environment. During storage, configuration and transportation the vote counter should be save. Before the counting process starts the feature to verify the authenticity of the vote counter will be used[5]. Specifically, it shall be ensured that access to the area where counting takes place is controlled. The electoral committee makes a count of the admitted voters and compare this to the number of votes cast in order to ensure that no additional votes have been counted. Further a number of random ballot papers are checked for proper counting. This way a significant manipulation should be detected.
OE.Admin	It shall be ensured that the administrative roles that interact with the vote counter have been trained with respect to their responsibilities. However the vote counter operator shall not be skilled in detection of attempts of attacks on the vote counter or are able to detect that there is malfunction. Furthermore it is assumed, that storage and distribution of the tokens falls into the responsibility of an administrative role and that therefore, for the vote counter, it can be assumed that only persons that are allowed to have access to the tokens can have that access. Storage and distribution in this case refers on the one hand to the phase when an election is prepared and the tokens are distributed to the administrative roles that operate the TOE.

[5] The objectives regarding storage, configuration, transportation and the verification of authenticity are not realistic and enforceable (from a security point of view). These objectives in the current Protection Profile are necessary because there are no known other physical protection mechanisms to warrant the integrity of the hardware of the vote counter.

Objective for environment	Description
OE.Token	It shall be ensured that the token for administrative purposes are evaluated according to [PP_AM].
OE.SM	It shall be ensured that the TOE has a built-in security module that provides the required cryptographic functionality and that has been certified according to [PP_SM].

4.3 Security Objectives rationale

4.3.1 Overview

The following table gives an overview how the assumptions, threats, and organisational security policies are addressed by the security objectives. The text of the following sections justifies this more in detail.

	O.Process	O.Integrity	O.Log	O.Management	O.DataExchange	O.Selfprotection	O.AccessControl	OE.Replacement	OE.SecurityFeature	OE.Expendable	OE.Environment	OE.Admin	OE.Token	OE.SM
T.MultipleVotes	X					X					X	X		
T.WrongVote	X	X				X								
T.WrongPoll				X	X	X								
T.WithholdVote	X					X		X		X	X			
T.ManipulatedCounting		X			X	X					X			
T.Log			X		X	X	X							
T.UnauthorizedAdmin						X	X					X	X	
T.UnauthorisedUse						X	X					X		
T.WrongModeChange						X	X							
T.Hack						X								
T.System_Forgery						X					X			
T.IncorrectNumber		X				X								
OSP.Log			X			X								
A.Replacement								X						
A.SecurityFeature									X					
A.Expendable										X				
A.Environment											X			
A.Admin												X		
A.Token													X	
A.SM														X

Table 9: Rationale for Security Objectives

4.3.2 Countering the threats

The following sections provide more detailed information on how the threats are countered by the security objectives for the TOE and its operational environment.

4.3.2.1 General objectives

The security objectives **O.Selfprotection** contribute to counter each threat as it ensures the integrity of the TOE in a general sense.

O.Management is needed as it defines the requirements around the management of the Security

Functions. Without a secure management no TOE can be secure. Also **OE.Admin** contributes to this aspect as it provides the requirements on the availability of trustworthy roles. **O.Integrity** requires the TOE to protect data in terms of integrity. Relevant events will be audited according **O.Log** that enables control whether the TOE works as specified. **O.DataExchange** allows import and export of required data, while its integrity and authenticity is ensured by the TOE's digital signature. The signed export of the counting result contributes against the thread T.ManipulatedCounting after export. **O.AccessControl** ensures that only authorized roles are able to get access to the vote counter depending on its current mode.

Those general objectives that have been argued in the previous paragraphs will not be addressed in detail in the following paragraphs.

4.3.2.2 T.MultipleVotes

The threat **T.MultipleChoices** is covered by a combination of the security objectives **O.Process, O.SelfProtection**, **OE.Environment** and **OE.Admin**.

O.Process requires the TOE to generate a reliable and correct result of the number of counted votes on the ballot papers and therefore counters this threat. **O.Selfprotection** ensures that the TOE cannot be manipulated without detection to count a single scanned ballot paper more than one time. **OE.Environment** and **OE.Admin** should ensure that the operators of the TOE are trustworthy.

4.3.2.3 T.WrongVote

The threat **T.WrongVote** is covered by a combination of the security objectives **O.Process, O.Integrity** and **O.Selfprotection**.

O.Process requires the TOE to generate a reliable and correct result of the number of counted votes on the ballot papers and therefore counters this threat. **O.Integrity** is responsible to protect the result of the counted ballot papers while they are kept in the TOE and **O.Selfprotection** ensures that the result of the counted ballot papers cannot be manipulated by an attack against the hardware or the software.

4.3.2.4 T.WrongPoll

The threat **T.WrongPoll** is covered by a combination of the security objectives **O.Management**, **O.DataExchange** and **O.Selfprotection**.

O.Selfprotection ensures that the election data cannot be manipulated by unauthorised users without detection. **O.Management** restricts the access to the management functionality of the TOE and the token that actives the functionality to configure the election data to authorized persons. **O.DataExchange** ensures that only data with verifiable integrity and authenticity can be imported into the TOE.

4.3.2.5 T.WithholdVote

The threat **T.WithholdVote** is covered by a combination of the security objectives **O.Process**, **O.Selfprotection, OE.Replacement, OE.Expendable** and **OE.Environment.**

O.Process and **OE.Environment** should ensure that ballot papers will be scanned. Further, in **OE.Environment** it is defined that the Electoral Committee compares the total number of votes cast with the number of admitted voters. **O.Selfprotection** ensures that the TOE cannot be manipulated without detection to withhold votes. **OE.Replacement** and **OE.Expendable** ensure that spare vote counters as well as used materials are available at an adequate amount for the case that the vote counter becomes un-operational or the vote counter runs out of material like ink or papers .

4.3.2.6 T.ManipulatedCounting

The threat **T.ManipulatedCounting** is covered by a combination of the security objectives **O.Integrity**, **O.Selfprotection, O.DataExchange** and **OE.Environment.**

O.Integrity and **O.Selfprotection** ensure that the TOE cannot be manipulated without detection in a way that the counting results are manipulated. **O.DataExchange** ensures that the TSF data that is used in the context of counting is transmitted in a protected manner. Further, in **OE.Environment** it is

483 defined that the Electoral Committee compares the total number of votes cast with the number of
484 admitted voters.

485 #### 4.3.2.7 T.Log

486 The threat **T.Log** is covered by a combination of the security objectives **O.Log**, **O.DataExchange,**
487 **O.Selfprotection** and **O.AccessControl.**

488 **O.Selfprotection** ensures that the log in the vote counter cannot be manipulated without detection.
489 **O.Log** and **O.AccessControl** ensure that only authorized roles have access to the log and that every
490 action is recorded with integrity. **O.DataExchange** requires that exported audit records must be signed
491 to ensure its integrity and authenticity.

492 #### 4.3.2.8 T.UnauthorizedAdmin

493 The threat **T.UnauthorizedAdmin** is covered by a combination of the security objectives
494 **O.Selfprotection O.AccessControl**, **OE.Admin** and **OE.Token**.

495 **O.Selfprotection** ensures that the vote counter cannot be manipulated without detection to use
496 administrative functionalities outside the specification. **O.AccessControl** and **OE.Admin** ensure that
497 users can only gain access to the functionalities that they are allowed to use. **OE.Token** requires the
498 use of tokens that have been evaluated in accordance with [PP-AM] and must therefore ensure a high
499 security against manipulation.

500 #### 4.3.2.9 T.UnauthorisedUse

501 The threat **T.UnauthorizedUsed** is covered by a combination of the security objectives
502 **O.Selfprotection O.AccessControl** and **OE.Admin.**

503 **O.Selfprotection** ensures that the vote counter cannot be manipulated without detection to enable the
504 counting by persons without a token. **O.AccessControl** and **OE.Admin** ensure that a user only gains
505 access with the token to the functionalities they are allowed to use in a specific mode of the TOE.

506 #### 4.3.2.10 T.WrongModeChange

507 The threat **T.WrongModeChange** is covered by a combination of the security objectives
508 **O.Selfprotection** and **O.AccessControl.**

509 **O.Selfprotection** ensures that the vote counter cannot be manipulated without detection to make a
510 mode change that is not allowed and gain access to functionalities that should not be available.
511 **O.AccessControl** enforces that only persons that are represented by dedicated token can change the
512 mode of the TOE and have no access to modes that should not be available.

513 #### 4.3.2.11 T.Hack

514 The threat **T.Hack** is covered by the security objective **O.Selfprotection.**
515 **O.Selfprotection** ensures that the vote counter is protected against vulnerabilities to
516 compromise or exploit a vote counter.

517 #### 4.3.2.12 T.System_Forgery

518 The threat **T.System_forgery** is covered by the security objectives **O.Selfprotection** and
519 **OE.Environment.**
520 **O.Selfprotection** ensures that parts of the vote counter cannot be manipulated without a
521 detection. The feature to verify authenticity makes it possible to detect a non authentic vote
522 counter. **OE.Environment** ensures that the feature to verify the authenticity of the vote
523 counter is used before the counting process starts.

524 #### 4.3.2.13 T.IncorrectNumber

525 The threat **T.IncorrectNumber** is covered by a combination of the security objectives **O.Integrity**
526 and **O.Selfprotection.**

527 **O.Integrity** and **O.Selfprotection** ensure that the consecutive number printed on the ballot papers
528 cannot be manipulated while the ballot papers are processed within the TOE.

529 ### 4.3.3 Coverage of organisational security policies

530 The following sections provide more detailed information about how the security objectives for the
531 environment and the TOE cover the organizational security policies.

532 #### 4.3.3.1 OSP.Log

533 The Organisational Security Policy **OSP.Log** that mandates that the TOE maintains an audit log is
534 directly addressed by the security objective for the TOE **O.Log**

535 ### 4.3.1 Coverage of assumptions

536 The following sections provide more detailed information about how the security objectives for the
537 environment cover the assumptions.

538 #### 4.3.1.1 A.Replacement

539 The assumption **A.Replacement** is directly and completely covered by the security objective
540 **OE.Replacement**. The assumption and the objective for the environment are drafted in a way that the
541 correspondence is obvious.

542 #### 4.3.1.2 A.SecurityFeature

543 The assumption **A.SecurityFeature** is covered by the security objective **OE.SecurityFeature**. The
544 assumption and the objective for the environment are drafted in a way that the correspondence is
545 obvious.

546 #### 4.3.1.3 A.Expendable

547 The assumption **A.Expandable** is directly and completely covered by the security objective
548 **OE.Expandable**. The assumption and the objective for the environment are drafted in a way that the
549 correspondence is obvious.

550 #### 4.3.1.4 A.Environment

551 The assumption **A.Environment** is directly and completely covered by the security objective
552 **OE.Environment**. The assumption and the objective for the environment are drafted in a way that the
553 correspondence is obvious.

554 #### 4.3.1.5 A.Admin

555 The assumption **A.Admin** is directly and completely covered by the security objective **OE.Admin**.
556 The assumption and the objective for the environment are drafted in a way that the correspondence is
557 obvious.

558 #### 4.3.1.6 A.Token

559 The assumption **A.Token** is directly and completely covered by the security objective **OE.Token**. The
560 assumption and the objective for the environment are drafted in a way that the correspondence is
561 obvious.

562 #### 4.3.1.7 A.SM

563 The assumption **A.SM** is directly and completely covered by the security objective **OE.SM**. The
564 assumption and the objective for the environment are drafted in a way that the correspondence is
565 obvious.

5 Extended Component definition

5.1 Definition of the Family ALC_DEL.2

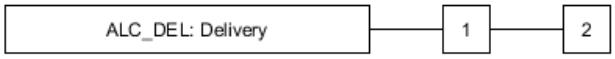
Objectives

The concern of this family is the secure transfer of the finished TOE from the development environment into the responsibility of the user.

The requirements for delivery call for system control and distribution facilities and procedures that detail the measures necessary to provide assurance that the security of the TOE is maintained during distribution of the TOE to the user. For a valid distribution of the TOE, the procedures used for the distribution of the TOE address the objectives identified in the PP/ST relating to the security of the TOE during delivery.

The extension of this family shall ensure the qualification of every single vote counter. This means that every device shall be investigated after its production whether it corresponds to the evaluated version of the TOE. The investigation shall ensure, that the developer has not changed or modified any component.

Component levelling



This family contains two components. An increasing level of protection is established by requiring commensurability of the delivery procedures with the assumed attack potential in the family Vulnerability analysis (AVA_VAN).

Application notes

Transportations from subcontractors to the developer or between different development sites are not considered here, but in the family Development security (ALC_DVS).

The end of the delivery phase is marked by the transfer of the TOE into the responsibility of the user. This does not necessarily coincide with the arrival of the TOE at the user's location.

The delivery procedures should consider, if applicable, issues such as:

a) ensuring that the TOE received by the consumer corresponds precisely to the evaluated version of the TOE;

b) avoiding or detecting any tampering with the actual version of the TOE;

c) preventing submission of a false version of the TOE;

d) avoiding unwanted knowledge of distribution of the TOE to the consumer: there might be cases where potential attackers should not know when and how it is delivered;

e) avoiding or detecting the TOE being intercepted during delivery; and

f) avoiding the TOE being delayed or stopped during distribution.

The delivery procedures should include the recipient's actions implied by these issues. The consistent description of these implied actions is examined in the Preparative procedures (AGD_PRE) family, if present.

The description of **ALC_DEL.2** refers to the terms “user” and “consumer”. Within this document, these terms are synonym to the governmental agency that receives the produced vote counters. It has been balanced whether it was better to develop a new assurance component or to use a known

606 component and augment it. The latter has been chosen due to the assumption, that it is more suitable
607 for evaluation if dedicated components base on the existing structure of classes and families.

608 **ALC_DEL.2 Delivery procedures**

Dependencies:	No dependencies.
	Developer action elements:
ALC_DEL.2.1D	The developer shall document and provide procedures for delivery of the TOE or parts of it to the consumer.
ALC_DEL.2.2D	The developer shall use the delivery procedures.
ALC_DEL.2.3D	**The developer shall document and provide evidence that every single vote counter corresponds precisely to the evaluated version of the TOE.**
Content and presentation elements:	
ALC_DEL.2.1C	The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to the consumer.
Evaluator action elements:	
ALC_DEL.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ALC_DEL.1.2E	**The evaluator shall confirm for every single vote counter that it corresponds precisely to the evaluated version of the TOE.**

609

610 5.2 Definition of the Family FDP_FPC.1

611 Objectives

612 The concern of the family FDP_FPC (Functional Process Controls) is the insurance that functional
613 processes performed by the TOE operate exactly as defined in dedicated rules and policies.

614 The family “Functional Process Controls” (FDP_FPC) is specified as follows

615 **Family behaviour:**

616 This family defines the functional processes the TOE has to provide.

617 **Component levelling:**

Management:	FDP_FPC.1 There are no management activities foreseen.
Audit:	The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST: a) Basic: Violation of rules or policies

618 **FDP_FPC.1 Functional Process Controls**

FDP_FPC.1.1	The TSF shall enforce the [*assignment: process policy*] for the following [*assignment: list of subjects, information, objects, processes*].
FDP_FPC.1.2	The TSF shall enforce the following rules [*assignment: process rules*].
Hierarchical to:	No other components
Dependencies:	No other components

6 Security Requirements

6.1 Overview

This chapter describes the security functional and the assurance requirements which have to be fulfilled by the TOE. Those requirements comprise functional components from part 2 of [CC] and the assurance components as defined for the Evaluation Assurance Level 4 from part 3 of [CC].

The following notations are used:

- **Refinement** operation (denoted by **bold text**): is used to add details to a requirement, and thus further restricts a requirement. In case that a word has been deleted from the original text this refinement is indicated by ~~**crossed out bold**~~ text
- **Selection** operation (denoted by underlined text): is used to select one or more options provided by the [CC] in stating a requirement.
- **Assignment** operation (denoted by *italicised text)*: is used to assign a specific value to an unspecified parameter, such as the length of a password.
- **Iteration** operation: are identified with a suffix in the name of the SFR (e.g. FMT_MOF.1/Mode).

It should be noted that the requirements in the following chapters are not necessarily be ordered alphabetically. Where useful the requirements have been grouped.

The following table summarises all TOE security functional requirements of this PP:

Class FAU: Security Audit	
FAU_ARP.1	Security alarms
FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAA.1	Potential violation analysis
FAU_STG.1	Protected audit trail storage
FAU_STG.4	Prevention of audit data loss
Class FCS: Cryptographic Support	
FCS_COP.1	Cryptographic operation
Class FDP: User Data Protection	
FDP_ACC.2	Complete access control
FDP_ACF.1	Security attribute based access control
FDP_DAU.1	Basic Data Authentication
FDP_FPC.1	Functional Process Controls
FDP_IFC.2	Complete information flow control
FDP_IFF.1	Simple security attributes
FDP_ITT.2	Transmission separation by attribute
FDP_ITT.4	Attribute-based integrity monitoring

FDP_SDI.2	Stored data integrity monitoring and action
Class FIA: Identification and Authentication	
FIA_AFL.1	Authentication failure handling
FIA_ATD.1	User attribute definition
FIA_UAU.2	User authentication before any action
FIA_UID.2	User identification before any action
FIA_USB.1	User-subject binding
Class FMT: Security Management	
FMT_MTD.1	Management of TSF data
FMT_MOF.1	Management of security functions behaviour
FMT_MOF.1/Mode	Management of security functions behaviour for the mode
FMT_MSA.3	Static attribute initialisation
FMT_MSA.1	Management of security attributes
FMT_MSA.2	Secure security attributes
FMT_SMR.1	Security roles
FMT_SMF.1	Specification of Management Functions
Class FPT: Protection of the TSF	
FPT_PHP.2	Notification of physical attack
FPT_PHP.3	Resistance to physical attack
FPT_RCV.3	Automated recovery without undue loss
FPT_RCV.4	Function recovery
FPT_TST.1	TSF testing
FPT_ITI.1	Inter-TSF detection of modification
FPT_RPL.1	Replay detection
FPT_FLS.1	Failure with preservation of secure state
FPT_STM.1	Reliable time stamps
Class FRU: Resource utilisation	
FRU_FLT.2	Limited fault tolerance
Class FTA: TOE access	
FTA_SSL.3	TSF-initiated termination
FTA_SSL.4	User-initiated termination
FTA_TAB.1	Default TOE access banners

FTA:TAH.1	TOE access history
FTA_TSE.1	TOE session establishment

637 **Table 10: List of Security Functional Requirements**

638 6.2 Class FAU: Security Audit

639 6.2.1.1 Security audit automatic response (FAU_ARP)

640 6.2.1.1.1 FAU_ARP.1: Security alarms

FAU_ARP.1.1 The TSF shall take [*notify the user and enter the mode "management"*] upon detection of a potential security violation.

Hierarchical to: No other components

Dependencies: FAU_SAA.1

641 6.2.1.2 Security audit data generation (FAU_GEN)

642 6.2.1.2.1 FAU_GEN.1: Audit data generation for system log

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events for the [detailed] level of audit; and

c) [*additional audit events for all actions performed by the TOE as specified in Table 11,*

d) [assignment: further actions or none]]

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [assignment*: other audit relevant information*].

Hierarchical to: No other components

Dependencies: FPT_STM.1

Application Note: The following table lists relevant events for the level of audit "detailed" structured after all used SFRs.

643

644

645
646

SFR	Audited events
FAU_ARP.1	Actions taken due to potential security violations.
FAU_GEN.1	-
FAU_GEN.2	-
FAU_SAA.1	Enabling and disabling of any of the analysis mechanisms; Automated responses performed by the tool.
FAU_STG.1	Actions taken due to exceeding of a threshold.
FAU_STG.4	Actions taken due to the audit storage failure.
FCS_COP.1	Any applicable cryptographic mode(s) of operation, subject attributes and object attributes.
FDP_ACC.2	-
FDP_ACF.1	The specific security attributes used in making an access check.
FDP_DAU.1	The identity of the subject that requested the evidence.
FDP_FPC.1	Violation of rules or policies
FDP_IFC.2	-
FDP_IFF.1	-
FDP_ITT.2	All attempts to transfer user data, including the protection method used and any errors that occurred.
FDP_ITT.4	The action taken upon detection of an integrity error.
FDP_SDI.2	The type of integrity error that occurred. The action taken upon detection of an integrity error.
FIA_AFL.1	The reaching of the threshold for the unsuccessful authentication attempts and the actions (e.g. disabling of a terminal) taken and the subsequent, if appropriate, restoration to the normal state (e.g. re-enabling of a terminal).
FIA_UAU.2	All use of the authentication mechanism.
FIA_USB.1	Success and failure of binding of user security attributes to a subject (e.g. success or failure to create a subject).
FIA_ATD.1	-
FIA_UID.2	All use of the user identification mechanism, including the user identity provided.
FMT_MTD.1	All modifications to the values of TSF data.
FMT_MOF.1	All modifications in the behaviour of the functions in the TSF.
FMT_MOF.1/Mode	All modifications in the behaviour of the functions in the TSF.
FMT_MSA.3	Modifications of the default setting of permissive or restrictive rules.

SFR	Audited events
	All modifications of the initial values of security attributes.
FMT_MSA.1	All modifications of the values of security attributes.
FMT_MSA.2	All offered and rejected values for a security attribute; All offered and accepted secure values for a security attribute.
FMT_SMR.1	Every use of the rights of a role.
FMT_SMF.1	Use of the management functions.
FPT_PHP.2	Detection of intrusion.
FPT_PHP.3	-
FPT_TST.1	Execution of the TSF self tests and the results of the tests.
FPT_RCV.3	Type of failure or service discontinuity
FPT_RCV.4	If possible, the detection of a failure of a function.
FPT_ITI.1	Detected modification of TSF data during transmission and action taken.
FPT_RPL.1	Action to be taken based on the specific actions.
FPT_FLS.1	Failure of the TSF.
FPT_STM.1	Providing a timestamp.
FRU_FLT.2	Any failure detected by the TSF.
FTA_SSL.3	Termination of an interactive session by the session locking mechanism.
FTA_SSL.4	Termination of an interactive session by the user.
FTA_TAB.1	-
FTA_TAH.1	-
FTA_TSE.1	Capture of the value of the selected access parameters (e.g. location of access, time of access).

647 **Table 11: Audit events**

Event	Additional information
Update software/firmware code	Token ID
Perform selftest	Token ID
Import of election configuration data	Token ID
Test vote counter function	Token ID
Export of the log file	Token ID
Export of counting result	Token ID
Erase of counting results, configuration data and log	Token ID
Import of token data	Token ID

Event	Additional information
Import of key store configuration data	Token ID
Enter number of polling station or ballot box number	Token ID
Export election configuration data	Token ID
Export token data	Token ID
Export firmware/software	Token ID
Activation with token	Token ID
Change of mode	Token ID
Count ballots	Token ID
Print counting result	Token ID
Recognised Vote	Token ID
Error that has occurred, like out of paper, paper jam, wrong token for current mode, not-authentic token used	Token ID

Table 12: Additional Audit events

6.2.1.2.2 FAU_GEN.2: Audit data generation for system log

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

Hierarchical to: No other components

Dependencies: FAU_GEN.1
FIA_UID.1

Application Note: It should be noted that the system of authentication of the TOE bases on tokens. Those tokens are treated as users even though the TOE will never get hold of the real user identity. Whenever the identity of the user is mentioned in the context of an SFR, this therefore refers to the ID of the token.

6.2.1.2.3 FAU_SAA.1 Potential violation analysis

FAU_SAA.1.1 The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the enforcement of the SFRs.

FAU_SAA.1.2 The TSF shall enforce the following rules for monitoring audited events:

a. Accumulation or combination of [assignment: *subset of defined auditable events*] known to indicate a potential security violation;

b. [assignment: *any other rules*].

Hierarchical to: No other components

Dependencies:	FAU_GEN.1
Application Note:	The accumulation of events that has to be filled into the assignment in FAU_SAA.1.2 strongly depends on the concrete implementation of the TOE. It is therefore left open to the specification and ST author.

652 #### 6.2.1.3 Security audit event storage (FAU_STG)

653 ##### 6.2.1.3.1 FAU_STG.1 Protected audit trail storage

FAU_STG.1.1	The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.
FAU_STG.1.2	The TSF shall be able to [prevent] unauthorised modifications to the stored audit records in the audit trail.
Hierarchical to:	No other components
Dependencies:	FAU_GEN.1

654 ##### 6.2.1.3.2 FAU_STG.4: Prevention of audit data loss

FAU_STG.4.1	The TSF shall [ignore audited events] and [*switch into the mode "management"*] if the audit trail is full.
Hierarchical to:	FAU_STG.3
Dependencies:	FAU_STG.1
Application Note:	Before the audit trail is full the TOE must give warnings.

655
656

657 6.3 Class FCS: Cryptographic Operation

658 ##### 6.3.1.1.1 FCS_COP.1 Cryptographic operation

FDP_COP.1.1	The TSF shall perform [*hashing, signature verification*] in accordance with a specified cryptographic algorithm [*assignment: cryptographic algorithm*] and cryptographic key sizes [*assignment: cryptographic key sizes*] that meet the following: [*assignment: list of standards*].
Hierarchical to:	No other components.
Dependencies:	FDP_IFF.1 Simple security attributes
Application Note:	The signature algorithm, cryptographic algorithm and cryptographic key sizes will have to be agreed with the responsible public authority.

659

660 ## 6.4 Class FDP: User data protection

661 ### 6.4.1.1 Access control policy (FDP_ACC)

662 #### 6.4.1.1.1 FDP_ACC.2: Complete access control

FDP_ACC.2.1 The TSF shall enforce the [*vote counter access SFP*] on [

Subjects:

- all users
- *[assignment: list of further subjects, or none]*.

Objects:

- *vote,*
- *persistent and ephemeral vote counter data,*
- *all TSF data,*
- *[assignment: list of further objects, or none]*.

] and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

Hierarchical to: FDP_ACC.1

Dependencies: FDP_ACF.1

Application Note: The SFR FDP ACC.2 introduces the access control policy for the TOE. A more functional overview over this can be found in chapter 1.4.8

The TOE refers to the current mode of operation and the role of the current user for access control. In so far the access control functionality can be seen as a special form of a Role Based Access Control.

More details on the rules that are used for access control can be found in FDP_ACF.1.

663 ### 6.4.1.2 Access control functions (FDP_ACF)

664 #### 6.4.1.2.1 FDP_ACF.1: Security attribute based access control

FDP_ACF.1.1 The TSF shall enforce the [*vote counter access SFP*] to objects based on the following: [

Security attributes for subjects:

- *Authenticated role of current user (ROLE_ID),*
- *Current mode (MODE_ID)*
- [*assignment: additional security attributes for subjects, or none*]

Security attributes for objects:

- [*assignment: additional security attributes for objects, or none*]

].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [

An operation between a subject and an object shall be allowed if

A) the ROLE_ID has the permission to perform this operation (as depicted in Table 13) AND

B) The operation is permitted within the current mode (MODE_ID) (as depicted in Table 13)

Else

The operation is prohibited

].

FDP_ACF.1.3	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [*none*].
FDP_ACF.1.4	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [*none*].
Hierarchical to:	No other components
Dependencies:	FDP_ACC.1 FMT_MSA.3
Application Note:	FDP ACF.1 defines the access control policy for the TOE. As outlined in chapter 1.4.8 it bases on the role of the current user and the current mode of the TOE. The access control policy rules as defined in FDP_ACF.1.2 ensure that an operation is only allowed if the role has the permission and the functionality is available in the current mode. By using "none" in the assignments in FDP_ACF.1.3 and FDP_ACF.1.4 it is ensured that the ST author cannot define additional rules that would overrule this access control policy.

665

TOE mode	**Role (ROLE_ID)**	**Allowed Operations**	**Possible next mode(s)[6]**
MANAGEMENT	(de)Configurator	Import election configuration data Test vote counter function	-
		Update software/ firmware code	-
		Import Token data Import key store configuration data	-
		Enter number of polling station or ballot box number	-

[6] A mode is identified by its MODE_ID

		Export log Export election configuration data Export token data Export counting result Export firmware/software	-
		Erase counting data, configuration data and log	-
		Shutting down system	-
		Change mode, only possible if configuration data is complete, tokens have been assigned to elections and number of polling station or ballot box has been entered	ELECTION
	Vote Counter Operator	Shutting down system	-
ELECTION	(de)Configurator	None	-
		Enter / verify number of polling station or ballot box number Perform selftest	-
		Count ballots Print counting result	-
		Export counting result	-
	Vote Counter Operator	Shutting down system	MANAGEMENT
	–	Detection of a possible manipulation or a defect	MANAGEMENT

666 **Table 13: TOE modes and subjects allowed interaction in the mode**

667 **6.4.1.3 Data authentication (FDP_DAU)**

668 **6.4.1.3.1 FDP_DAU.1: Basic Data Authentication**

FDP_DAU.1.1 The TSF shall provide a capability to generate evidence that can be used as a guarantee of the validity of [*the log and counting result*].

FDP_DAU.1.2 The TSF shall provide [*the (de)configurator*] with the ability to verify evidence of the validity of the indicated information.

Hierarchical to: No other components

Dependencies:	No dependencies
Application Note:	FDP_DAU.1 is present in this PP to make sure that the log file and counting result that can be exported from the TOE is authentic and integer. Such functionality can e.g. be implemented by the use of a digital signature. Such a signature would then allow the reviewer to verify that the log file is authentic and integer.

669 **6.4.1.4 Functional Process Controls (FDP_FPC)**

670 **6.4.1.4.1 FDP_FPC.1 Functional Process Controls**

FDP_FPC.1.1	The TSF shall enforce the [*assignment: process policy*] for the following [*assignment: list of subjects, information, objects, processes*].
FDP_FPC.1.2	The TSF shall enforce the following rules [*assignment: process rules*].
Hierarchical to:	No other components
Dependencies:	No other components
Application Note:	This TSF models the concrete rules that model the counting functionality of the TOE. This needs to be in the PP to have security objective O.Process covered. The concrete assignments in FDP_FPC.1 are left open to the ST author. The ST author shall consider the corresponding functional specification for the vote counter when completing the assignments in FDP_FPC.1.

671 **6.4.1.5 Information flow control policy (FDP_IFC)**

672 **6.4.1.5.1 FDP_IFC.2 Subset information flow control**

FDP_IFC.2.1 The TSF shall enforce the [*internal information flow control SFP*] on [

Subjects: TOE modules

Information (assets):

- *logs,*
- *token data,*
- *configuration data,*
- *votes,*
- *counting data,*
- *ephemeral vote counter data,*

Operations: any

] and all operations that cause that information to flow to and from subjects covered by the SFP.

FDP_IFC.2.2 The TSF shall ensure that all operations that cause any information in the TOE to flow to and from any subject in the TOE are covered by an information flow control SFP.

Hierarchical to:	FDP_IFC.1
Dependencies:	FDP_IFF.1 Simple security attributes

673 **6.4.1.6 Information flow control functions (FDP_IFF)**

674 **6.4.1.6.1 FDP_IFF.1 Simple security attributes**

FDP_IFF.1.1	The TSF shall enforce the [*internal information flow control SFP*] based on the following types of subject and information security attributes: [*subjects and information according to FDP_IFC.2.1 and the following security attribute:* • *necessity to transfer the asset to other TOE modules*].
FDP_IFF.1.2	The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [*Any information listed in FDP_IFC.2.1 shall only be transferred between those TOE modules that actually need to process the information to fulfill their purpose according to the design of the TOE. If at any time such information is part of a larger set of information, TOE modules shall make sure to decompose the larger set and only transfer the necessary information to other TOE modules.*].
FDP_IFF.1.3	The TSF shall enforce ~~**the**~~ [*no further rules*].
FDP_IFF.1.4	The TSF shall explicitly authorise an information flow based on the following rules: [*none*].
FDP_IFF.1.5	The TSF shall explicitly deny an information flow based on the following rules: [*none*].
Hierarchical to:	No other components.
Dependencies:	FDP_IFC.1 Subset information flow control ~~**FMT_MSA.3 Static attribute initialisation**~~
Application Note:	FDP_IFC.2 and FDP_IFF.1 are used to express the requirement that the TOE assets shall not be available to all parts of the TOE but only to those parts that make use of it. The restriction on information flow defined in FDP_IFF.1.2 will ensure that. FDP_IFF.1.3, FDP_IFF.1.4 and FDP_IFF.1.5 are not used because there are no further rules necessary to express the requirement. In this case, according to [CC Part 2, chapter F.6], the PP/ST author should specify “none”. Since the security attribute *necessity to transfer the asset to other TOE modules* is determined during development for each asset and is not configurable, the dependency FMT_MSA.3 of FDP_IFF.1 is not necessary. TOE modules and their interactions will be described in detail by the developer to fulfil the requirements of ADV_TDS.5. Therefore, the evaluator has all means to verify the correct implementation of this SFP. During evaluation of aspect ADV INT.3 the evaluator will also analyze whether the modular design of the TOE is well-structured. A well-

structured modular design supports that sensitive information is only present where necessary.

675 **6.4.1.7 Internal TOE transfer (FDP_ITT)**

676 **6.4.1.7.1 FDP_ITT.2 Transmission separation by attribute**

FDP_ITT.2.1	The TSF shall enforce the [*vote counter access SFP or internal information flow control SFP*] to prevent the [modification and loss of use] of user data when it is transmitted between physically-separated parts of the TOE.
FDP_ITT.2.2	The TSF shall separate data controlled by the SFP(s) when transmitted between physically-separated parts of the TOE, based on the values of the following: [assignment: security attributes that require separation].
Hierarchical to:	FDP_ITT.1
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]

677 **6.4.1.7.2 FDP_ITT.4 Attribute-based integrity monitoring**

FDP_ITT.4.1	The TSF shall enforce the [*vote counter access SFP or internal information flow control SFP*] to monitor user data transmitted between physically-separated parts of the TOE for the following errors: [assignment: integrity errors], based on the following attributes: [assignment: security attributes that require separate transmission channels].
FDP_ITT.4.2	Upon detection of a data integrity error, the TSF shall [assignment: specify the action to be taken upon integrity error].
Hierarchical to:	FDP_ITT.3
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FDP_ITT.2 Transmission separation by attribute
Application Note:	It should be noted that the requirements FDP_ITT.2 and FDP_ITT.4 are dedicated to cases in which the TOE comprises physically separated parts. In cases, where the TOE does not comprise physically separated parts, those requirements shall be considered being fulfilled without any implementation/evidence.

678

679 **6.4.1.8 Stored data integrity (FDP_SDI)**

680 **6.4.1.8.1 FDP_SDI.2 Stored data integrity monitoring and action**

FDP_SDI.2.1	The TSF shall monitor user data stored in containers controlled by the TSF for [*integrity errors*] on all objects, based on the following attributes: [assignment: *attributes defined by the ST author*].
FDP_SDI.2.2	Upon detection of a data integrity error, the TSF shall [*switch into the mode "management"*].
Hierarchical to:	FDP_SDI.1
Dependencies:	No dependencies

Application Note: The user data controlled by the TSF (votes, counting data, ephemeral data) must have attributes that enable the TOE to monitor the integrity of this data. The attribute may be a suitable hash value or any other suitable attribute that matches the specification and has to be specified by the ST author in the ST in the last assignment in FDP_SDI.2.1.

6.5 Class FIA: Identification and Authentication

Application Note: The concept to operate the TOE is based on a procedure that activates the TOE for a specific purpose. This activation uses digital token that are presented to the TOE and are dedicated to a specific role (see Table 13) with a limited functionality and only in dedicated modes. More precisely: Every role has a specific token and is only able to activate the TOE for their specific purpose if the TOE is in a mode where this role is allowed to interact with the TOE. For more details on the access control policy behind this concept please refer to chapter 1.4.8.

Please note that even though the SFRs within this chapter refer to a "user" this does not mean that the identity of the user has to be known by the TOE.

6.5.1.1 Authentication failures (FIA_AFL)

6.5.1.1.1 FIA_AFL.1 Authentication failure handling

FIA_AFL.1.1 The TSF shall detect when [selection: [assignment: *positive integer number*], an administrator configurable positive integer within [assignment*: range of acceptable values*]] unsuccessful authentication attempts occur related to [assignment*: list of authentication events*].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been [selection: *met, surpassed*], the TSF shall [assignment: *list of actions*].

Hierarchical to: No other components.

Dependencies: FIA_UAU.1 Timing of authentication

Application Note: FIA_AFL.1 is used in this PP to ensure that the authentication functionality is resistant against brute force attacks. It is in the intention of the authors of this PP that the mechanism behind it shall only block the authentication function of the TOE for a certain amount of time after a certain number of unsuccessful attempts occurred. This way is can be ensured that this function cannot be misused to attack the availability of the TOE. However, the concrete assignments in FIA_AFL.1 are left to the specification and ST author as they highly depend on implementation details (such as the speed of the authentication function)

6.5.1.2 Token attribute definition (FIA_ATD)

6.5.1.2.1 FIA_ATD.1 User attribute definition

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: [*role-id, token-id [assignment: additional security attributes, or none]*].

Hierarchical to: No other components.

Dependencies: No dependencies.

687 #### 6.5.1.3 User identification (FIA_UID)

688 **6.5.1.3.1 FIA_UID.2 User identification before any action**

FIA_UID.2.1 The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Hierarchical to: FIA_UID.1

Dependencies: No dependencies

689 #### 6.5.1.4 User authentication (FIA_UAU)

690 **6.5.1.4.1 FIA_UAU.2 User authentication before any action**

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Hierarchical to: FIA_UAU.1

Dependencies: FIA_UID.1

691 #### 6.5.1.5 User-subject binding (FIA_USB)

692 **6.5.1.5.1 FIA_USB.1 User-subject binding**

FIA_USB.1.1 The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [*role-id, token-id, current mode [assignment: additional security attributes, or none]*].

FIA_USB.1.2 The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [*assignment: rules for the initial association of attributes*].

FIA_USB.1.3 The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [*no changes of the security attributes are allowed during a session*].

Hierarchical to: No other components.

Dependencies: FIA_ATD.1

Application Note: The initial rules for the association of attributes to the subjects depend on the concrete implementation. Therefore, the assignment in FIA_USB.1.2 is left to the specification and ST author. In any case it has to be ensured that the binding of attributes happens directly after the user (more precisely: the token of the user) has been identified and authenticated.

693 ## 6.6 Class FMT: Security Management

694 #### 6.6.1.1 Management of TSF data (FMT_MTD)

695 **6.6.1.1.1 FMT_MTD.1 Management of TSF data**

FMT_MTD.1.1 The TSF shall restrict the ability to [*import, export and delete as depicted in Table 13*] the [*all TSF data*] to [*roles that are associated with modes as depicted in Table 13*].

Hierarchical to: No other components

Dependencies:	FMT_SMR.1 FMT_SMF.1
Application Note:	The TOE shall control access to the TSF data to authorized roles within dedicated modes. This means that the TOE has a predefined mode changes and within each mode only dedicated roles are allowed to manage the TSF data. The assignment of roles to modes is shown in Table 13.

696 **6.6.1.2 Management of security attributes (FMT_MSA)**

697 **6.6.1.2.1 FMT_MSA.3: Static attribute initialisation**

FMT_MSA.3.1	The TSF shall enforce the [*vote counter access SFP, information flow control SFP*] to provide [*restrictive*] default values for security attributes that are used to enforce the SFP.
FMT_MSA.3.2	The TSF shall allow ~~the~~ [*nobody*] to specify alternative initial values to override the default values when an object or information is created.
Hierarchical to:	No other components
Dependencies:	FMT_MSA.1 FMT_SMR.1

698 **6.6.1.2.2 FMT_MSA.1: Management of security attributes**

FMT_MSA.1	The TSF shall enforce the [vote counter *access control SFP(s)*] to restrict the ability to [*modify*] the security attributes [*all security attributes*] to [*no role*].
Hierarchical to:	No other components
Dependencies:	FDP_ACC.1 FMT_SMR.1 FMT_SMF.1

699 **6.6.1.2.3 FMT_MSA.2 Secure security attributes**

FMT_MSA.2.1	The TSF shall ensure that only secure values are accepted for [*all security attributes and TSF data*].
Hierarchical to:	No other components
Dependencies:	FDP_ACC.1 FMT_MSA.1 FMT_SMR.1
Application Note:	The TOE shall ensure that only secure values are accepted for all security attributes. This is specifically (but not only) the case for all data that is imported from outside the scope of control of the TOE. This requirement specifically applies to the configuration data, token data and the software/firmware updates that must only be accepted and processed by the TOE if the attached signatures can be verified. It is acknowledged that the possibility of the TOE to ensure that only secure values for TSF data in general are accepted is limited.

700 **6.6.1.3 Security management roles (FMT_SMR)**

701 **6.6.1.3.1 FMT_SMR.1: Security roles**

FMT_SMR.1.1	The TSF shall maintain the roles [• *(de)configurator,* • *Vote counter operator*].
FMT_SMR.1.2	The TSF shall be able to associate users with roles.
Hierarchical to:	No other components
Dependencies:	FIA_UID.1

702 **6.6.1.4 Specification of Management Functions (FMT_SMF)**

703 **6.6.1.4.1 FMT_SMF.1 Specification of Management Functions**

FMT_SMF.1.1	The TSF shall be capable of performing the following management functions: [• *activation of a mode of operation ("change mode")* • *import election configuration data* • *export election configuration data* • *import token data,* • *import key store configuration data* • *export token data,* • *update firmware/software* • *export log,* • *erase configuration data,* • *erase counting data, configuration data and log,* • *export counting result,* • *export firmware/software,* • *perform selftest,* • *test vote counter function,* • *enter number of polling station or ballot box number,* • *enter / verify number of polling station or ballot box number* • *[assignment: additional management functions, or none]]*.
Hierarchical to:	No other components
Dependencies:	No dependencies
Application Note:	It should be noted that the access to the management functionality as defined in FMT_SMF.1 is restricted to certain administrative roles. The restriction of access is defined in the SFRs of the families FMT_MOF (see below) and the SFRs for access control.

704 #### 6.6.1.5 Management of functions in TSF (FMT_MOF)

705 ##### 6.6.1.5.1 FMT_MOF.1 Management of security functions behaviour

FMT_MOF.1.1	The TSF shall restrict the ability to [*modify the behaviour of*] the functions [*all management functions*] to [*nobody*].
Hierarchical to:	No other components
Dependencies:	FMT_SMR.1 FMT_SMF.1

706 ##### 6.6.1.5.2 FMT_MOF.1/Mode Management of security functions behaviour for the mode of
707 operation

FMT_MOF.1.1	The TSF shall restrict the ability to [**change**] the ~~**functions**~~ [*mode of operation*] to [*roles and modes as depicted in* Table 13].
Hierarchical to:	No other components
Dependencies:	FMT_SMR.1 FMT_SMF.1
Application Note:	The mode of operation for the TOE is an essential aspect of the access control policy of the TOE. Therefore, FMT_MOF.1/Mode has been introduced in order to make sure that only users of authorized roles are allowed to change the mode. More details on the restrictions can be found in Table 13.

708 ## 6.7 Class FPT: Protection of the TSF

709 #### 6.7.1.1 TSF physical protection (FPT_PHP)

710 ##### 6.7.1.1.1 FPT_PHP.2: Notification of physical attack

FPT_PHP.2.1	The TSF shall provide unambiguous detection of physical tampering that might compromise the TSF.
FPT_PHP.2.2	The TSF shall provide the capability to determine whether physical tampering with the TSF's devices or TSF's elements has occurred.
FPT_PHP.2.3	For [*the vote counter and its casing*], the TSF shall monitor the device~~s~~ ~~**and elements**~~ and notify [*all roles*] when physical tampering with the TSF's devices or TSF's elements has occurred.
Hierarchical to:	FPT_PHP.1
Dependencies:	FMT_MOF.1
Application Note:	Based on the assumption that the vote counter operator is not trained in the detection of tampering, the self-protection mechanism of the TOE will detect intrusion and switch the TOE automatically into the mode “management”. The (de)configurator is allowed to print and export count results, export logs, configuration data, token data and firmware/software of the TOE for investigation.

711 **6.7.1.1.2 FPT_PHP.3: Resistance to physical attack**

FPT_PHP.3.1 The TSF shall resist [

- *physical tampering attacks*
- *[assignment*: *physical tampering scenarios or none*]

to the [*casing of the TOE [assignment: list of TSF elements or none]*] by responding automatically such that the SFRs are always enforced.

Hierarchical to: No other components

Dependencies: No dependencies

712 **6.7.1.2 Trusted recovery (FPT_RCV)**

713 **6.7.1.2.1 FPT_RCV.3 Automated recovery without undue loss**

FPT_RCV.3.1 When automated recovery from *[assignment: list of failures/service discontinuities or none]* is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.3.2 For [*power blackout, [assignment: list of failures/service discontinuities or none]*], the TSF shall ensure the return of the TOE to a secure state using automated procedures.

FPT_RCV.3.3 The functions provided by the TSF to recover from failure or service discontinuity shall ensure that the secure initial state is restored without exceeding [assignment: *quantification*] for loss of TSF data or objects under the control of the TSF.

FPT_RCV.3.4 The TSF shall provide the capability to determine the objects that were or were not capable of being recovered.

Hierarchical to: No other components

Dependencies: No dependencies

Application Note: The concrete assignments in FPT_RCV.3 are left to the ST author. When completing them, the ST author shall consider the functional specification of the vote counter as well as security aspects (e.g. it might be helpful to define that the current user will have to get re-authenticated after a relevant failure).

714 **6.7.1.2.2 FPT_RCV.4 Function recovery**

FPT_RCV.1.1 The TSF shall ensure that [*all functions*] have the property that the function either completes successfully, or for the indicated failure scenarios, recovers to a consistent and secure state.

Hierarchical to: No other components

Dependencies: No dependencies

Application Note: Secure state in this context means that the TOE shall either resume or abort the counting process. For the case that a function recovery in this sense is not possible the TOE shall fall to its mode “management”. In this way it can be ensured that the TOE never operates within an undefined state.

715 **6.7.1.3 TSF self test (FPT_TST)**

716 **6.7.1.3.1 FPT_TST.1: TSF testing**

FPT_TST.1.1 The TSF shall run a suite of self tests [*during initial start-up, periodically during normal operation, at the request of the authorised user, at the conditions [assignment: conditions under which self test should occur or none]]* to demonstrate the correct operation of [the TSF].

FPT_TST.1.2 The TSF shall provide authorised users with the capability to verify the integrity of [

- *logs,*
- *configuration data,*
- *token data,*
- *software/firmware of the TOE*
- *[assignment: parts of the TOE or none]].*

FPT_TST.1.3 The TSF shall provide authorised users with the capability to verify the integrity of [

- *the internal hardware,*
- *the internal software/firmware*
- *the printing unit,*
- *the casing,*
- *the interfaces,*
- *[assignment: parts of TSF or none]]*.

Hierarchical to: No other components

Dependencies: No dependencies

Application Note: The verification of the integrity of software/firmware may be implemented in software or hardware like a Trusted Platform Module (TPM). This implementation is part of the TOE and therefore part of the evaluation of the TOE. Verification of software/firmware relies on the integrity of the hardware. Therefore the mechanism of verifying the integrity of the hardware needs to be reliable and trustworthy.

717 **6.7.1.4 Integrity of exported TSF data (FPT_ITI)**

718 **6.7.1.4.1 FPT_ITI.1 Inter-TSF detection of modification**

FPT_ITI.1.1 The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and another trusted IT product within the following metric: [assignment: *a defined modification metric*].

FPT_ITI.1.2	The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and another trusted IT product and ~~**perform**~~ [*switch the TOE into the mode "blocked"*] if modifications are detected.
Hierarchical to:	No other components
Dependencies:	No dependencies
Application Note:	Electoral data, referendum data, the log file and the counting result are data can be exchanged between the TOE and other digital products that provide this data or related to the log file where they are stored to. In the context of FPT_ITI.1 these digital products are trusted IT products and it has to be ensured that modification on the data during the transmission will be detected.

719 **6.7.1.5 Replay detection (FPT_RPL)**

720 **6.7.1.5.1 FPT_RPL.1: Replay detection**

FPT_RPL.1.1	The TSF shall detect replay for the following entities: [*ballot papers that have been scanned more than once in the same counting sequence* .
FPT_RPL.1.2	The TSF shall ~~**perform**~~ [*put the ballot paper into the dedicated output tray*] when replay is detected.
Hierarchical to:	No other components
Dependencies:	No dependencies

721 **6.7.1.6 Fail secure (FPT_FLS)**

722 **6.7.1.6.1 FPT_FLS.1: Failure with preservation of secure state**

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: [• *the self-test detects an error or manipulation,* • *the self-protection detects a manipulation* • *[assignment, other failures to be defined by the ST author]*].
Hierarchical to:	No other components
Dependencies:	No dependencies
Application Note:	The secure state mentioned in the SFR FPT_FLS.1 refers to the mode "management" within the life-cycle model.

723 **6.7.1.7 Time stamps (FPT_STM)**

724 **6.7.1.7.1 FPT_STM.1 Reliable time stamps**

FPT_STM.1.1	The TSF shall be able to provide reliable time stamps.
Hierarchical to:	No other components

Dependencies: No dependencies

6.8 Class FRU: Resource utilisation

6.8.1.1 Fault tolerance (FRU_FLT)

6.8.1.1.1 FRU_FLT.2 Limited fault tolerance

FRU_FLT.2.1 The TSF shall ensure the operation of all the TOE's capabilities when the following failures occur: [*assignment: list of type of failures*].

Hierarchical to: FRU_FLT.1

Dependencies: FPT_FLS.1

6.9 Class FTA: TOE access

6.9.1.1 Session locking and termination (FTA_SSL)

6.9.1.1.1 FTA_SSL.3 TSF-initiated termination

FTA_SSL.3.1 The TSF shall terminate an interactive session after a [*assignment: time interval of user inactivity*].

Hierarchical to: No other components

Dependencies: No dependencies

Application Note: The assignment in FTA_SSL.3.1 allows specifying the time after which the TOE shall end the session with a user. This time interval highly depends on the concrete implementation of the TOE and is therefore left to the specification and ST author.

6.9.1.1.2 FTA_SSL.4 User-initiated termination

FTA_SSL.4.1 The TSF shall allow user-initiated termination of the user's own interactive session.

Hierarchical to: No other components

Dependencies: No dependencies

6.9.1.2 TOE access banners (FTA_TAB)

6.9.1.2.1 FTA_TAB.1 Default TOE access banners

FTA_TAB.1.1 Before establishing a user session, the TSF shall display an advisory warning message regarding unauthorised use of the TOE.

Hierarchical to: No other components

Dependencies: No dependencies

735 **6.9.1.3 TOE access history (FTA_TAH)**

736 **6.9.1.3.1 FTA_TAH.1 TOE access history**

FTA_TAH.1.1	Upon successful session establishment **for a (de)configurator**,, the TSF shall display the [*date, time, method, location*] of the last successful session establishment to the user.
FTA_TAH.1.2	Upon successful session establishment **for a (de)configurator**,, the TSF shall display the [*date, time, method, location*] of the last unsuccessful attempt to session establishment and the number of unsuccessful attempts since the last successful session establishment.
FTA_TAH.1.3	The TSF shall not erase the access history information from the user interface without giving the user an opportunity to review the information.
Hierarchical to:	No other components
Dependencies:	No dependencies
Application Note:	The TOE access history only applies to logins of a (de)configurator. Each (de)configurator shall be presented the last successful and unsuccessful login attempts of this administrative role.

737 **6.9.1.4 TOE session establishment (FTA_TSE)**

738 **6.9.1.4.1 FTA_TSE.1 TOE session establishment**

FTA_TSE.1.1	The TSF shall be able to deny session establishment based on [*the assignment of roles to dedicated modes as outlined in Table 14*].
Hierarchical to:	No other components
Dependencies:	No dependencies
Application Note:	The interaction of the

739

	(de)Configurator	**Vote Counter Operator**
Management	X	X
Election	-	X

740 Table 14: TSF managing subjects and the modes they have access to the TOE

741

6.10 Security Assurance Requirements for the TOE

The minimum Evaluation Assurance Level for this Protection Profile is **EAL 4 augmented** by **ALC_DVS.2**, **AVA_VAN.5** and the use of the explicit component **ALC_DEL.2**.

The following table lists the assurance components which are therefore applicable to this PP.

Assurance Class	Assurance Component
Development	ADV_ARC.1
	ADV_FSP.4
	ADV_IMP.1
	ADV_TDS.3
Guidance documents	AGD_OPE.1
	AGD_PRE.1
Life-cycle support	ALC_CMC.4
	ALC_CMS.4
	ALC_DEL.2
	ALC_DVS.2
	ALC_LCD.1
	ALC_TAT.1
Security Target Evaluation	ASE_CCL.1
	ASE_ECD.1
	ASE_INT.1
	ASE_OBJ.2
	ASE_REQ.2
	ASE_SPD.1
	ASE_TSS.1
Tests	ATE_COV.2
	ATE_DPT.1
	ATE_FUN.1
	ATE_IND.2
Vulnerability Assessment	**AVA_VAN.5**

Table 15: Assurance Requirements

6.11 Security Requirements rationale

6.11.1 Security Functional Requirements rationale

6.11.1.1 Fulfilment of the Security Objectives

This chapter proves that the set of security requirements (TOE) is suited to fulfil the security objectives described in chapter 4 and that each SFR can be traced back to the security objectives. At least one security objective exists for each security requirement.

	O.Process	O.Integrity	O.Log	O.Management	O.DataExchange	O.Selfprotection	O.AccessControl
FAU_ARP.1	X					X	
FAU_GEN.1			X				
FAU_GEN.2			X				
FAU_SAA.1	X					X	
FAU_STG.1			X				
FAU_STG.4			X				
FCS_COP.1		X					
FDP_ACC.2							X
FDP_ACF.1							X
FDP_DAU.1		X	X				
FDP_FPC.1	X						
FDP_IFC.2						X	
FDP_IFF.1						X	
FDP_ITT.2						X	
FDP_ITT.4						X	
FDP_SDI.2		X	X				
FIA_AFL.1							X
FIA_ATD.1							X
FIA_UID.2							X
FIA_UAU.2							X
FIA_USB.1							X
FMT_MTD.1				X	X		

	O.Process	O.Integrity	O.Log	O.Management	O.DataExchange	O.Selfprotection	O.AccessControl
FMT_MSA.1				X			
FMT_MSA.2						X	
FMT_MSA.3				X			
FMT_SMR.1							X
FMT_MOF.1				X			
FMT_MOF.1/Mode							X
FMT_SMF.1				X			
FPT_PHP.2						X	
FPT_PHP.3						X	
FPT_RCV.3	X	X					
FPT_RCV.4	X						
FPT_TST.1	X					X	
FPT_ITI.1					X		
FPT_RPL.1	X	X					
FPT_FLS.1	X					X	
FPT_STM.1			X				
FRU_FLT.2	X					X	
FTA_SSL.3							X
FTA_SSL.4							X
FTA_TAB.1							X
FTA_TAH.1							X
FTA_TSE.1							X

754 **Table 16: Fulfilment of Security Objectives**

755 The following paragraphs contain more details on this mapping.

756 **6.11.1.1.1 O.Process**

757 O.Process is met by a combination of the following SFRs:

- **FDP_FPC.1** defines how the TOE shall process the ballot papers. Thus, this SFR ensures that the overall process of the scanning is performed in a way that meets the requirements of O.Process.
- **FAU_ARP.1 and FAU_SAA.1** should ensure that security relevant events that could have an impact to the secure counting process will be detected.
- **FPT_RCV.3** defines the requirements for automated recovery in case of certain errors during the counting process and therefore supports the secure counting process.
- **FPT_RCV.4** ensures that the TOE is able to recover to a secure state in case processing exceptions are encountered and therefore supports a robust counting process.
- **FPT_TST.1** defines a self test that helps to ensure that the security features of the TOE (including the counting functionality) are working correctly.
- **FPT_RPL.1** defines the requirement that the TOE shall detect replayed ballot paper and therewith supports the secure counting process.
- **FPT_FLS.1** and **FRU_FLT.2** define how the TOE shall react in case of errors and support that the security functionality will work as specified.

6.11.1.1.2 O.Integrity

O.Integrity is met by a combination of the following SFRs:

- **FDP_DAU.1** provides the functions to verify integrity and authenticity of the log and counting result.
- **FDP_SDI.2** ensures that it the integrity of the counting data is monitored and that the vote counter will change its mode to “management” in case of integrity failures.
- **FPT_RCV.3** requires that the TOE is able to recover to a secure state without loss of data or at least the possibility to detect the data it was not capable to restore. This ensures the integrity of the counting in case of unexpected power blackouts.
- **FPT_RPL.1** requires that the TOE is able to detect ballot papers that already have been scanned within in the same counting sequence. This contributes to the integrity of the counting result.
- **FCS_COP.1** provides the functionality of hashing and verification of digital signatures to verify the integrity of imported data. The hashing and verification of digital signatures allows the detection of any manipulation of the imported and signed data and contributes therefore to the protection of the integrity of this data.

6.11.1.1.3 O.Log

O.Log is met by a combination of the following SFRs:

- **FAU_GEN.1** and **FAU_GEN.2** define that a log file must be generated and define the records that shall be audited.
- **FAU_STG.1** ensures that the audit records cannot be manipulated and deleted from unauthorised roles and contributes therefore to the availability of the log.
- **FAU_STG.4** defines the behaviour if the audit trail is full and ensures that no audit data is lost.
- **FDP_DAU.1** provides the functions to verify integrity and authenticity of the log and counting result and thus the possibility to ensure that the log and counting result have not been manipulated.
- **FDP_SDI.2** defines requirements on the integrity protection for data, including the log file.
- **FPT_STM.1** provides the time that can be used by the audit functionality to provide the events with a timestamp. Those timestamps allow the tracing of entities and their actions with the vote counter

804 **6.11.1.1.4 O.Management**

805 O.Management is met by a combination of the following SFRs:

806 • **FMT_MTD.1** defines the roles that are allowed to manage TSF data and defines the actions
807 those roles are allowed to perform with the TOE. This ensures that the access should be
808 limited to the functionalities for which the role is authorized.

809 • **FMT_MSA.1** ensures that no role should be able to change the security attributes and can
810 cause vulnerabilities of the TOE due to configuration errors or attacks.

811 • **FMT_MSA.3** defines the initialization attributes and that no role should be able to change the
812 default values. This ensures that the TOE always uses valid default values on its start-up.

813 • **FMT_MOF.1** ensures that a role cannot change the behaviour of security functions. Similar to
814 FMT_MSA.1 this should ensure that misconfiguration do not lead to any vulnerabilities of the
815 TOE.

816 • **FMT_SMF.1** defines the management functions that the TOE shall provide. This ensures that
817 the TOE does not provide any functionality that is not necessary and could lead to a lack of
818 security.

819 **6.11.1.1.5 O.DataExchange**

820 O.DataExchange is met by a combination of the following SFRs:

821 • **FMT_MTD.1** defines the roles that are allowed execute data exchange. This ensures that only
822 allowed roles are able to import or export data and prevents that other roles may use this
823 functionality to import/export data.

824 • **FPT_ITI.1** defines the requirements for detections of modifications when data is submitted
825 between the TOE and another trusted IT product (which is the core functionality as required
826 by O.DataExchange).

827 **6.11.1.1.6 O.Selfprotection**

828 O.Selfprotection is met by a combination of the following SFRs:

829 • **FAU_ARP.1** ensures that the TOE notifies the user if it detects a security violation. This
830 should ensure that the user is informed in case of a potential security violation.

831 • **FAU_SAA.1** requires that the TOE is able to analyze its audited events and should therefore
832 be capable to detect a potential security violation based on these records.

833 • **FDP_ITT.2** and **FDP_ITT.4** ensure secure handling of data when transmitted between
834 physically separated parts of the TOE.

835 • **FMT_MSA.2** ensures the acceptance of secure values. This is specifically relevant when data
836 is imported from outside the scope of control of the TOE and therewith adds to the self
837 protection capabilities as required by this objective.

838 • **FPT_PHP.2** defines the requirements for the physical protection that the TOE must provide
839 and the behaviour of the TOE if it detects tampering. This should ensure that a physical
840 tampering attack to the TOE its hardware and its casing will lead to an action defined in
841 FAU_PHP.3 and FAU_ARP.1.

842 • **FPT_PHP.3** defines an automated response to tampering scenarios. This should ensure that
843 the TOE will be able to react in an adequate manner if it detects physical tampering attacks

844 • **FPT_TST.1** defines allowed self-testing functionality to check the correct working of the
845 TOE. Such a self-test should be able to detect manipulation of hardware, software, data, the
846 connection of fake devices and the manipulation of the power supply.

847 • **FPT_FLS.1** defines that in case of errors or tampering the TOE will switch to a secure state
848 (the mode to “management”).

849 • **FRU_FLT.2** ensures that the TOE can react in tolerance to a number of well-defined error
850 states. This enhances the self-protection capabilities of the TOE.

- **FDP_IFC.2** and **FDP_IFF.1** ensure that sensitive information is only transferred between those parts of the TOE that actually need it. This helps to protect the TOE against attacks that try to recover sensitive information.

6.11.1.1.7 O.AccessControl

O.AccessControl is met by a combination of the following SFRs:

- **FDP_ACC.2** and **FDP_ACF.1** define the vote counter access SFP. This SFP ensures that only the defined roles within dedicated modes should have access to the TOE and to the functionality.
- **FIA_AFL.1** ensures that the authentication mechanism should be protected against brute force attacks.
- **FIA_ATD.1** defines the security attributes to be assigned to a token. These attributes are necessary to implement the access policies of roles to functions of the TOE.
- **FIA_UAU.2 and FIA_UID.2** requires that every entity must be successfully authenticated and identified before that entity can perform an action with the TOE. This should ensure that an entity is not able to perform an action without permission.
- **FIA_USB.1** defines the mapping between security attributes and subjects to enforce the access SFP.
- **FMT_SMR.1** defines the security roles used by the TOE.
- **FMT_MOF.1/Mode** ensures that changing the mode of operation is limited to certain roles and based on the current mode.
- **FTA_SSL.3, FTA_SSL.4** and **FTA_TSE.1** require and define a session based access control that is used to grant access to the TOE.
- **FTA_TAB.1** ensures that the TOE presents the user with the access banners that are defined in O.AccessControl.
- **FTA_TAH.1** ensures that the TOE presents the (de)configurator with information about their last successful and unsuccessful login attempts after they successfully logged in

6.11.1.2 Fulfilment of the dependencies

The following table summarises all TOE functional requirements dependencies of this PP and demonstrates that they are fulfilled.

SFR	Dependencies	Fulfilled by
FAU_GEN.1	FPT_STM.1 Reliable time stamps	FPT_STM.1
FAU_STG.1	FAU_GEN.1 Audit data generation	FAU_GEN.1
FAU_STG.4	FAU_STG.1 Protected audit trail storage	FAU_STG.1
FDP_ACC.2	FDP_ACF.1 Security attribute based access control	FDP_ACF.1
FDP_IFC.2	FDP_IFF.1 Simple security attributes	FDP_IFF.1
FDP_IFF.1	FDP_IFC.1 Subset information flow control	FDP_IFC.2
	FMT_MSA.3 Static attribute initialisation	No component. Justification: The information flow control policy specified in FDP_IFF.1 and FDP_IFC.2 does

SFR	Dependencies	Fulfilled by
		not require to manage any security attributes.
FDP_ITT.2	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]	Both (depending on the implementation)
FDP_ITT.4	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FDP_ITT.2 Transmission separation by attribute	FDP ACC.2 and FDP_IFC.2 (depending on the implementation), FDP_ITT.2
FDP_ACF.1	FDP_ACC.1 Subset access control	FDP_ACC.2
	FMT_MSA.3 Static attribute initialisation	FMT_MSA.3
FIA_AFL.1	FIA_UAU.1 Timing of authentication	FIA_UAU.2
FIA_UAU.2	FIA_UID.1 Timing of identification	FIA_UID.2
FIA_USB.1	FIA_ATD.1 User attribute definition	FIA_ATD.1
FMT_MTD.1	FMT_MTD.1 Management of TSF data	FMT_MTD.1
	FMT_SMR.1 Security roles	FMT_SMR.1
FMT_MSA.3	FMT_MSA.1 Management of security attributes	FMT_MSA.1
	FMT_SMR.1 Security roles	FMT_SMR.1
FMT_MSA.1	FDP_ACC.1 Subset access control	FDP_ACC.2
	FMT_MSA.1 Management of security attributes	FMT_MSA.1
	FMT_SMR.1 Security roles	FMT_SMR.1
FMT_MSA.2	FDP_ACC.1 Subset access control	FDP_ACC.2
	FMT_MSA.1 Management of security attributes	FMT_MSA.1
	FMT_SMR.1 Security roles	FMT_SMR.1
FMT_SMR.1	FIA_UID.1 Timing of identification	FIA_UID.1
FMT_MOF.1	FMT_SMR.1 Security roles	FMT_SMR.1
	FMT_SMF.1 Specification of Management Functions	FMT_SMF.1
FMT_MOF.1	FMT_SMR.1 Security roles	FMT_SMR.1
	FMT_SMF.1 Specification of Management Functions	FMT_SMF.1
FPT_PHP.2	FMT_MOF.1 Management of security functions behaviour	FMT_MOF.1
FPT_RCV.3	AGD_OPE.1 Operational user guidance	AGD_OPE.1

881 **Table 17: SFR Dependencies**

6.11.1.3 Justification for missing dependencies

6.11.1.4 Justification for selection of assurance level

EAL4 permits a developer to maximise assurance gained from positive security engineering based on good commercial development practices. It is also the highest assurance level that enables the use of standard components (hardware and software). EAL4 is the highest level at which it is likely to be economically feasible to retrofit to an existing product line. It is applicable in those circumstances where developers or users require a moderate to high level of independently assured security in conventional commodity TOEs, and there is willingness to incur some additional security-specific engineering costs.

An EAL4 evaluation provides, in addition to EAL3, an analysis supported by a complete interface specification, a description of the basic modular design of the TOE, and a subset of the implementation. Testing is supported by a vulnerability analysis (also using the implementation representation), demonstrating resistance to penetration attackers with an Enhanced-Basic attack potential. Assurance is also provided through additional automated configuration management.

In addition to the measures that are included in the EAL4 package, three further components have been chosen in order to address dedicated aspects:

The assurance component ALC_DVS.2 provides evidence that security measures implement sufficient protection. This component will help assuring that security requirements are addressed in the design.

The explicit assurance component ALC_DEL.2 has been designed and selected in order to express a certain need in the context of the development and production of the vote counter. In standard evaluations it falls into the responsibility of the developer to ensure that each instance of the TOE that is produced matches the requirements from the specification and evaluation. In the context of the development of the criteria for the vote counter it became evident that this would not be sufficient in this context. Rather, a need has been identified that each instance of the TOE is checked after production in order to ensure that it needs the criteria. While this assurance requirement represents a significant effort it has been found that this is the only way to ensure that each and every vote counter that is used is secure and meets the requirements.

The augmentation by AVA_VAN.5 has been chosen to provide confidence that the TOE will resist sophisticated attacks.

6.11.2 Security Assurance Requirements rationale

6.11.2.1 Dependencies of assurance components

The dependencies of the assurance requirements taken from EAL 4 are fulfilled automatically. The augmentation by ALC_DEL.2, ALC_DVS.2 and AVA_VAN.5 does not introduce additional assurance components that are not contained in EAL 4.

916 7 Appendix

917 7.1 Glossary

Authenticity	Property that an entity is what it claims to be.
Authority for investigation	See chapter 3.1
Ballot paper	Special paper that is used to print the choices.
Choice	See chapter 3.2
Confidentiality	The property that information is not made available or disclosed to unauthorised individuals, entities, or processes.
Configuration data	See chapter 3.2
EAL	Evaluation Assurance Level
Electoral committee	See chapter 3.1
Ephemeral vote printer data	See chapter 3.2
Integrity	Property that sensitive data has not been modified or deleted in an unauthorised and undetected manner.
Logs	See chapter 3.2
Maintenance authority	See chapter 3.1
TOE	Target of Evaluation -set of software, firmware and/or hardware possibly
Token	In this context a hardware component that is used to switch between the modes and to activate the TOE.
Token data	See chapter 3.2
Vote counter reviewer	See chapter 3.1
Voter	See chapter 3.1

918 7.2 References

[CC] Common Criteria for Information Technology Security Evaluation –

- Part 1: Introduction and general model, dated September 2012, version 3.1, Revision 4
- Part 2: Security functional requirements, datedSeptember2012, version 3.1, Revision 4
- Part 3: Security assurance requirements, dated September 2012, version 3.1, Revision 4

[PP_AM] PP for the authentication module, equivalent to one of the following:
Protection profiles for secure signature creation device — Part 2: Device

	with key generation (BSI-CC-PP-0059-2009-MA-01) Protection profiles for secure signature creation device — Part 3: Device with key import (BSI-CC-PP-0075)
[PP_SM]	PP for the internal security module, equivalent to one of the following: Protection profiles for secure signature creation device — Part 2: Device with key generation (BSI-CC-PP-0059-2009-MA-01) Protection profiles for secure signature creation device — Part 3: Device with key import (BSI-CC-PP-0075)

919