9

Proof of Concept

The previous part discusses the GI/BSI/DFKI Protection Profile which constitutes after the implementation of the identified improvements as the proposed evaluation methodology for remote electronic voting systems. The result can now be applied to available systems. Currently, there is no system that has been evaluated against the GI/BSI/DFKI Protection Profile or even against the improved version.

This chapter aims to gain experiences with the application of this evaluation framework. Thus, the Estonian system and the POLYAS system[1] are analysed with respect to this framework. Due to space and time constrains, no complete Common Criteria evaluation has been undertaken. It has been decided to evaluate against the security problem definition retrieved from the extended and improved core Protection Profile as described in the previous chapter. This analysis is based on a system description deduced from available documents. The result is provided in this chapter.

9.1 Procedure Specification

Due to time and space constraints, no formal Common Criteria evaluation is presented for the Estonian system and the POLYAS system. The provided analysis is based on the security objectives from the GI/BSI/DFKI Protection Profile and the recommended extensions from Sect. 8.3, while, with respect to the secrecy of the vote, it is assumed that it is sufficient to ensure the secrecy of the vote till the next election (see Sect. 7.2.2 for further discussions). In addition, the analysis considers the PP assumptions about the environment

[1] A similar analysis has been done in joint work with Hugo Jonker for the Dutch Rijnland Internet Election System (RIES) in [76]. However, RIES does not fit to the considered target of evaluation from Sect. 6.2 because it provides voter verifiability and ensure the secrecy of the vote in the election setup phase. Thus, it is not further discussed here.

M. Volkamer: Evaluation of Electronic Voting, LNBIP 30, pp. 177–193, 2009.

and the intruder's technical capabilities as considered in Sect. 8.2.3. For each security objective, it is outlined whether (and if yes, with which TOE security function(s)) each of the two systems meets this particular security objective (in an adequate and sufficient way).

This analysis mainly corresponds to a Security Target evaluation (which is part of a Common Criteria evaluation). However, this analysis is based on the security objective, while a formal CC evaluation of the Security Target would be based on the security functional requirements. As Sect. 8.2.2 recommends using demonstrable conformance[2], this "easier" case is applied for the analysis.

Besides the main results (PASS, FAIL, and INCONCL[3]), the result ORG is applied to indicate that the developers are aware of corresponding problems or attacks but implement only organisational solutions to meet the corresponding security objective.

Requirements, which are extended, clarified, or added by the hints for improvements in Sect. 8.3, are labelled with an asterisk '*'. The detailed description of the processes and the protocols is separated from those during the election setup phase, those during the polling phase, and those during the result calculation process.

9.2 The Estonian System

Already in 2001, the Ministry of Justice announced intentions to introduce remote electronic voting. In 2005, remote electronic voting was implemented as an additional voting channel for local elections. Two years later, remote electronic voting for the Riigikogu (Estonian parliament) election was the first countrywide use of the Internet as a voting channel in a parliamentary election. There was no special registration process but each voter was able to vote using remote electronic voting. Even though there was no sign that the voters rejected remote electronic voting in the 2007 elections, only 5.4 percent of voters cast votes using the Internet as their voting channel.

According to the legislation, remote electronic voting is allowed under three main preconditions: firstly, the voter has to identify and authenticate himself with his digitally-enabled ID card[4], secondly, remote electronic voting is implemented as advance voting (from six to four days before election day), and thirdly, vote updating is enabled (in particular after having cast an electronic vote, the voter can overwrite this vote by casting a paper vote in an

[2] The necessary explanations as demanded for demonstrable conformance are left out.

[3] INCONCL means that the available sources do not provide enough information to determine any of the other verdicts.

[4] In Estonia, the new and already broadly distributed personal identification document (ID card) contains a chip which enables the user to be identified via the Internet and to digitally sign legally accepted documents.

advanced polling station). All technical activities related to the remote electronic voting process were audited by an external auditing company KMPG Baltics, including the election setup phase, polling phase, and tallying phase. The audit was performed against written documents describing the necessary steps and procedures.

The following system description and analysis are based on the following documents:

- OSCE/ODIHR Election Assessment Mission Report for the Parliamentary Elections of Estonia [106]
- The paper "Towards Remote E-Voting: Estonian case" [94]
- The paper "E-Voting in Estonia 2005. The First Practice of Country-wide Binding Internet Voting in the World" [95]

9.2.1 System Description

a) Classification

According to the classification from Sect. 2.1 the Estonian system can be classified in the following way:

- The Estonian System belongs to the *remote electronic voting system* category.
- The identification and authentication technique in use is a combination of *possession-based* and *secret-based*; in particular, the Estonian ID card is used, which identifies the voter over the Internet and enables the voter to digitally sign documents (for instance, his encrypted vote). To use this functionality, the voter needs to know his two PIN codes associated with the ID card: one for identification and one to sign documents.
- With respect to the secrecy of the vote the Estonian system is a representative of the class anonymity is ensured in the tallying phase by applying a hardware security module.
- The Estonian System belongs with respect to the different client-side voting software classes to the *fat-client* approach: there are three types of client-side voting software for the three different operating systems namely Windows, UNIX, and Apple MacOS.

The Estonian system does not match exactly the TOE description from Sect. 6.2 as it enables vote updating and allows the responsible election authority to change the electoral register (and in fact they did this every day). For the following analysis, this additional functionality is not considered.

b) Overview

From an abstract level, the Estonian system works in the following way: the voter logs onto the voting server and identifies himself with his ID card (using PIN1). Then, the voting server checks the voter's identity and provides the

corresponding ballot to the voter. After having made his choice, the voter digitally signs his encrypted vote. The voting server verifies the voter's signature. In the tallying phase, first, the digital signature is removed, then the encrypted votes are scrambled, and finally, they are decrypted by the HSM and counted. The main parties in the Estonian system are as follows:

- Registration server (RS)
- Certification server (CA)
- Vote storage server (VSS)
- Counting software on a separate PC (CPC)
- Hardware security module (HSM)
- Client-side voting software (CSS)

In the Estonian remote electronic voting system, the *one* voting server is further separated into three servers, namely the registration server, the vote storage server, and the certification server, while the last one is involved but not set up in particular for the remote electronic voting solution but for any application based on the digital identity card. The tallying software is partitioned in one part, running on the tallying PC and a second one running on the hardware security module to decrypt the votes. The Estonian System implements the following communication links:

- CSS - RS: to communicate with the remote electronic voting system
- RS - VSS: to forward votes to the storage
- VSS - CA: to check whether the voter's certificate is still valid.

c) Description of the Election Setup Phase

Preparation on the Server-Side. Preparation on the Server-side. On the server-side, several steps must be taken as follows: new RS, VSS, and CPC are purchased and reinstalled with an operating system, security mechanisms (for instance firewalls), and the corresponding voting/tallying software. The Hardware Security Module is set up; that is, a key pair is generated. While the secret one is stored on the device, the public key is integrated in the client-side voting software. Moreover, keys to enable the HSM are generated: seven keys, which are distributed to the National Election Commission (NEC) members, and two for the administrators. These keys are generated in a way that the two administration keys and four out of the seven NEC keys are necessary to enable the HSM.

Preparation on the Voter-Side. Preparation on the Voter-side. The voter needs to be prepared to use the electronic channel. Besides his electronic identity card, he needs to have a corresponding smart card reader and needs to know his two PINs. Moreover, if he uses MacOS or Linux, the voter needs to download the client-side voting software. In the case of a windows user, he needs to have Java enabled, so that the web browser can load the corresponding Java Applet.

d) Description of the Polling Phase

The high-level protocol steps during the polling phase are described in Fig. 9.1. This figure uses many shortcuts, therefore some explanations are given here:

- *SSL* – Two directed SSL connection (with the voter's first secret key enabled with PIN1).
- *elig*? - Here the RS checks whether the requesting person is an eligible voter.
- $re-vote$? - Here the VVS checks whether the requesting voter has already cast a vote[5].
- *gen ballot* - generate ballot that belongs to this particular voter.
- *choose* - the voter makes his choice.
- *vote* - the system displays the voter's choice and the voter verifies whether he wants to confirm this choice or changes his choice again.
- $sig(m)$ stands for signing the message m with the voter's secret key enabled with PIN2 while such a message is implicitly extended with the voter's certificate for the corresponding secret key.
- $enc(m)$ stands for encrypting a message m with the public key from the HSM.
- $sig-ID$ - the RS verifies whether the signature belongs to the person that started the session.
- *sig ok* - the VVS verifies the signature and the validity of the certificate.

e) Description of the Tallying Phase

After the electronic polling phase and closing the advanced polling stations, those e-votes stored at the VS where voters also cast a paper vote, are labelled with"not to be counted". Then a CD is burned containing the last received e-vote per voter in a randomised order (while those labelled with "not to be counted" are excluded). This CD is sealed and handed over to the NEC chairman. On election day, one hour before the polling stations close, the result calculation process starts. The e-votes are loaded on the CPC, which is connected to the HSM (via cable). Next, the HSM is enabled by entering four of the seven NEC keys and the two administration keys. Now the encrypted votes are sent to the HSM vote by vote, and the HSM sends corresponding decrypted votes back. Having finished the decryption, the votes are tallied and the result is burned onto another CD. This CD is loaded onto an other ordinary PC in order to display the result in a human readable way. The result is digitally signed by the NEC chairmen. The signed result is the legal one.

[5] In case, the voter has already cast an e-vote, this information is displayed to him and he is asked whether he wants to update his vote.

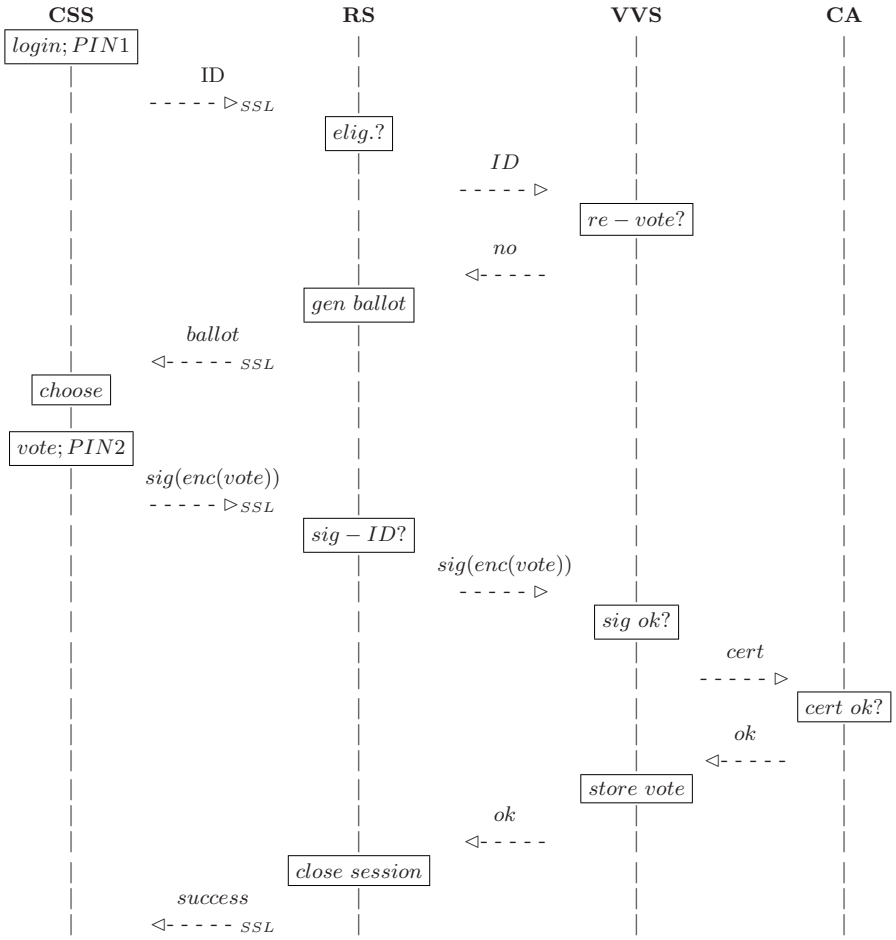

Fig. 9.1. The voting protocol implemented in the Estonian system

9.2.2 System Analysis

Based on this information, the identified security objectives are checked to see if they meet the requirements from chapter 8. The result of this evaluation is summarised in Table 9.1 (for O.T.) and 9.2 (for O.OSP.).

Result. The tables show that the Estonian system meets most of the security objectives with a PASS (at all 17). Two are met by organisational means and for seven of the security objectives no statement is possible due to missing information about the system. The inconclusive security objective only affect those objectives deduced from organisational security policies. As there is also no FAIL in the result, there is currently no reason that a formal Common Criteria evaluation of The Estonian System against the BSI/GI/DFKI Protection

Table 9.1. Result of the analysis for the Estonian system (part 2)

Security Objective	Result	Explanation
O.T.IneligVoter	PASS	The identification and authentication is based on the voter's digitally-enabled ID card.
O.T.OneVoterOneVote	PASS	The Estonian system implements vote updating, thus, Estonians are allowed to cast more than one vote. However, the system ensures that only the last vote is taken for the tallying. Note, this security objective would FAIL in the case of a strict conformance claim.
O.T.ProofGen	PASS	According to Fig. 9.1, it is not possible to generate a proof from any information either sent to, displayed on, and/or sent from a vote-casting device.
O.T.DeleteMsgNet	PASS	According to Fig. 9.1, this is ensured as long as the voter ensures that he receives the last confirmation.
O.T.AlterMsgNet	PASS	The communication is secured by SSL. In addition, votes are signed by the voter.
O.T.ElectionSecrecyNet	PASS	The communication is protected with SSL. In addition, votes are encrypted with the public key of the HSM.
O.T.IntResultNet	PASS	See O.T.ElectionSecrecyNet
O.T.WrongServer	PASS	As the Estonian system uses SSL, this security objective is ensured as long as the voter verifies the server certificate.
O.T.IntegElecData	ORG	After closing the poll, a CD is burned containing the last received e-vote per voter. This CD is sealed. Thus, the integrity of e-votes is only ensured by organisational means. In addition, the protected data only contains e-votes, while it is required to protect any kind of election data.
O.T.ElectionSecrecy	PASS	The encrypted e-votes are stored on the CD in a randomised order and without the voter's signature (anonymousness by scrambling the e-votes). After the tallying phase, there exists a second CD containing the list of decrypted votes. However, as it is not stored, it is unknown which encrypted e-vote from the first CD belongs to which voter, so O.ElectionSecrecy is ensured.
O.T.PersonalDataNet*	PASS	The identification data (ID) sent in the first steps of the protocol is secured with SSL.
O.T.SecretAuthNet*	PASS	It is not necessary to protect the authentication information on the Internet as only the voter can sign votes with his private key.

Table 9.2. Result of the analysis for the Estonian system (part 1)

Security Objective	Result	Explanation
O.OSP.Interface	PASS	All required functionality is implemented.
O.OSP.PWClosePoll	INCONCL	–
O.OSP.PWInterface	INCONCL	–
O.OSP.Confirmation	PASS	See Fig. 9.1.
O.OSP.SelfCheck	INCONCL	–
O.OSP.ErrorRecovery	INCONCL	–
O.OSP.Auditing	INCONCL	The Estonian system produces audit data, but it is not known which information is stored.
O.OSP.VoteRight	PASS	This is ensured by the implementation of vote updating.
O.OSP.VoteRightExc	INCONCL	–
O.OSP.SepDuty	PASS	Two administrators need to enter their passwords.
O.OSP.AC	ORG	There was an AC mechanism implemented on the voting server.
O.OSP.AccurCalc	PASS	This was shown by tests in advance of the election.
O.OSP.Feedback*	PASS	The administrators are informed via SMS.
O.OSP.NoInteract*	INCONCL	–

Profile should fail. In particular, there is no reason to change the architecture or the voting protocol in order to get the system certified. Minor changes with respect to the INCONCL security objectives might be necessary.

9.3 The POLYAS System

The POLYAS system is the voting system from a company called Micromata GmbH. It has a long-standing history – compared to the field itself – which starts in 1996, where the first election was carried out with 64.000 young Finnish pupils. Nowadays, the POLYAS system has been used to cast more than 340.000 votes, 210.000 of which were in Germany. In the last years, the system has been improved continuously by a close partnership with the Gesellschaft für Informatik (GI - the German society of computer scientists) and here the advisory board of security and voting experts. The GI has used the POLYAS system in parallel to postal voting for their yearly held elections since 2005. Beside several GI elections, the POLYAS system was also used for the elections of the Deutsche Forschungsgemeinschaft (DFG - German Research Foundation) in 2007.

For the system description and the analysis the following documents have been used:

- The paper [123][6] at the Vote-ID conference
- The POLYAS system Web page (www.polyas.de)
- Two confidential manufacturer's documents: one document describes how the POLYAS system ensures the requirements from the GI requirement catalogue [113], and the other one describes the procedure to activate the POLYAS system

The POLYAS system is described and analysed in the same software version as used for the GI elections.

9.3.1 System Description

a) Classification

According to the classification given in Sect. 2.1 the POLYAS system can be classified in the following way:

- The POLYAS system belongs to the *remote electronic voting systems* according to the defined election forms.
- The identification and authentication technique in use is *secret-based*[7]; in particular, the GI membership number is used to identify the voter and the authentication token is generated in the election setup phase and sent to the voter via ordinary mail.
- With respect to the secrecy of the vote, the POLYAS system is representative of the class *anonymity is ensured during the polling phase* and of the sub-class *separation of duty principle*.
- With respect to the different client-side voting software classes, the POLYAS system belongs to the *Web browser solution* approach; it supports any Web browser, including "lynx" a text-based Web browser.

b) Overview

The main parties in the POLYAS system are as follows:

- The client-side voting software (CSS)
- The electoral register server (ERS)
- The validator server (VS)
- The ballot box server (BBS)

In the POLYAS system, the *one* voting server is separated into three different servers, while each server is located at a different place and administrated by a different party.

[6] This paper is based on a contracted study developed by the e-voting.cc competence center and the author of this book.

[7] In the POLYAS implementation for the D21 elections, the identification and authentication technique is based on digital signature cards.

The POLYAS system implements the following communication links:

- CSS - ERS: to check the voting right,
- VS - ERS: to control the ERS's decision about the voter's voting right and to generate random anonymous authorisation tokens T,
- CSS - BBS: to cast a vote, and
- ERS - BBS: to inform the BBS about valid authorisation tokens and vise versa to inform the ERS about unauthorised tokens because corresponding votes have been cast.

c) Description of the Election Setup Phase

The election setup phase contains the following six main tasks:

1. Generation of the authentication token (TAN):
   The process[8] generating the TANs has as output only the hashed ($hash(TAN_i)$) and encrypted ($encr(sk_P, TAN_i)$) TANs.
   - ($hash(TAN_i)$) is linked to voter$_i$ in the electoral register (containing all membership numbers to identify the voter). This register is stored on the ERS.
   - ($encr(pk_P, TAN_i)$) is linked to voter $_i$ in another copy of the electoral register (containing the voter's addresses but not their membership numbers).
     This extended electoral register is sent to the provider. In order to prepare the election material for the voters, the provider decrypts the TANs with his secret key sk_P and prints the TANs on the election material. This material is sent to the voter.
2. The following three key pairs are generated per server:
   - Https key pair (only for ERS and BBS)
   - Communication key pair
   - Database key pair

   Each of the https secret keys is stored on the corresponding servers as well as corresponding public https keys from the other servers needed to later verify messages. The https public keys from the ERS and the BBS are made public to the voter (on the Web page and printed in the election material). The voter can use these two keys to later verify whether he communicates with the proper servers.
   The public and private communication and database keys are stored on the corresponding servers. The corresponding secret keys are encrypted with two pass phrases, in a way that both are necessary to decrypt the keys. The six pass-phrases are each known by one of the six different members of the responsible election authority.
3. In the electoral register (containing the membership numbers - ID), corresponding authentication tokens (TANs) are added in the following hashed and signed way:

[8] This process is not further discussed with respect to its security functions.

$$ID - -hash(TAN) - -sig_{ERS} - -sig_{VS}$$
where $sig_{ERS} := sig(sk_{ERS}, hash(TAN))$ and
$sig_{VS} := sig(sk_{VS}, sig_{ERS})$;
the secret keys are those from the communication key pair.

This electronic electoral register is stored on the ERS. It is also hashed, signed with sk_{ERS}, and then stored in a secure way outside the system.

4. The ballot is designed, and the rules to cast a valid vote are defined. Both sets of information are stored on the BBS.
5. Two access tokens are generated for each of the servers to get general access to the servers in order to, for instance, store the electoral register or start the polling phase. Again, here are six different secrets, which need to be distributed amongst the responsible election authority.
6. The servers are configured and secured (for instance, by installing a firewall and a virus scanner). Afterwards, the corresponding POLYAS Software for the particular server is installed.

The amount of keys and their distribution is shown in Fig. 9.2.

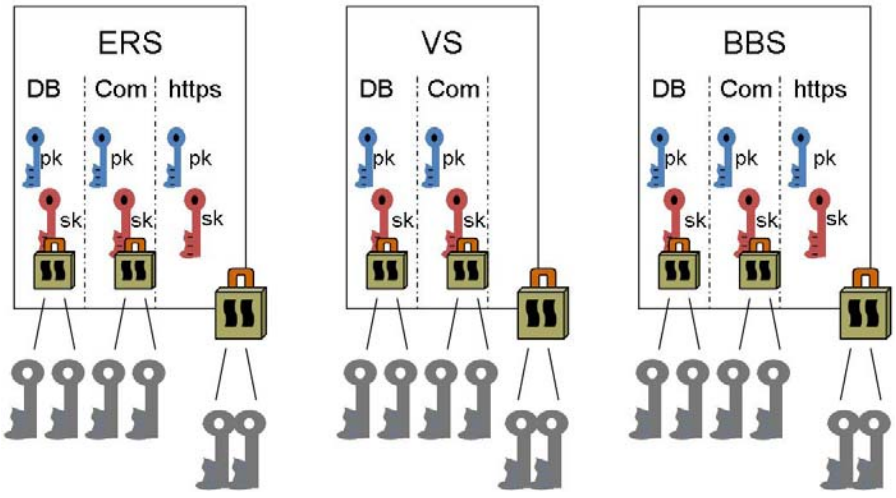

Fig. 9.2. Key distribution for the three POLYAS server

d) Description of the Polling Phase

First, the corresponding members of the responsible election authority log onto the BBS, using the authentication tokens in order to start the software. To do so, the other members responsible for this server need to enter their pass phrases to decrypt the database and the communication secret key (see Fig. 9.2). The same procedure needs to be taken for the VS. When both servers and the software run successfully, the pair knowing the access tokens for the

ERS uses these pass codes to log on and to start this software. Again, to do so, the other members involved need to enter their pass phrases to decrypt the database and the communication secret key. At the official beginning of the election, they start the polling phase using corresponding functionality of the ERS based on the POLYAS software.

The high-level protocol steps during the polling phase are described in Fig. 9.3. This figure uses many shortcuts, therefore some explanations are given here:

- SSL_i highlights all SSL communications of one session. In order to successfully cast a vote, four sessions are necessary (between different voting server components).
- *elig*? - Here the ERS checks whether the TAN corresponds to the ID and whether the corresponding voter has not cast a vote, yet.
- *checksig* verifies both received signatures (signed messages).
- *gen.* - means generate.
- T stands for the generated random authorisation token, which enables the voter to communicate anonymous with the BBS.
- *setinval.* - with this function the VS labels the value sig_{ERS} as invalid, that is, if there will be a second request from the ERS for a particular voter, the VS cancels the protocol (and in particular does not generate a new TAN T).
- *choose* - the voter makes his choice.
- *accept* - the voter confirms his choice (for the first time).
- *vote* - the system displays the voter's choice and the voter verifies whether he wants to confirm this choice or changes the choice again.
- *vote* := *choice* - In step *store* the voter's choice has already been stored in a database. In this step this choice is labelled as vote. At the end, only labelled database entries are tallied.

In addition, the communication between the servers is secured by SSL using the corresponding communication keys. All votes and voting tokens are stored in an encrypted and signed manner, using the public key of the involved database. Moreover, votes are stored in a randomised order in blocks of 30. As soon as one block is completed, the corresponding votes are concatenated to one string, which is hashed and published. The next block will be treated similarly but built as a hash-chain:

$$hash(permut(vote_1, .., vote_{30}))$$
$$hash(hash(permut(vote_{31}, .., vote_{60}))\#hash(permut(vote_1, .., vote_{30})))$$
$$...$$

e) Description of the Tallying Phase

To close the election, again the members of the responsible election authority who have the authentication tokens need to log onto the corresponding servers (see Fig. 9.2). In the next step, first the ERS is taken off-line followed by

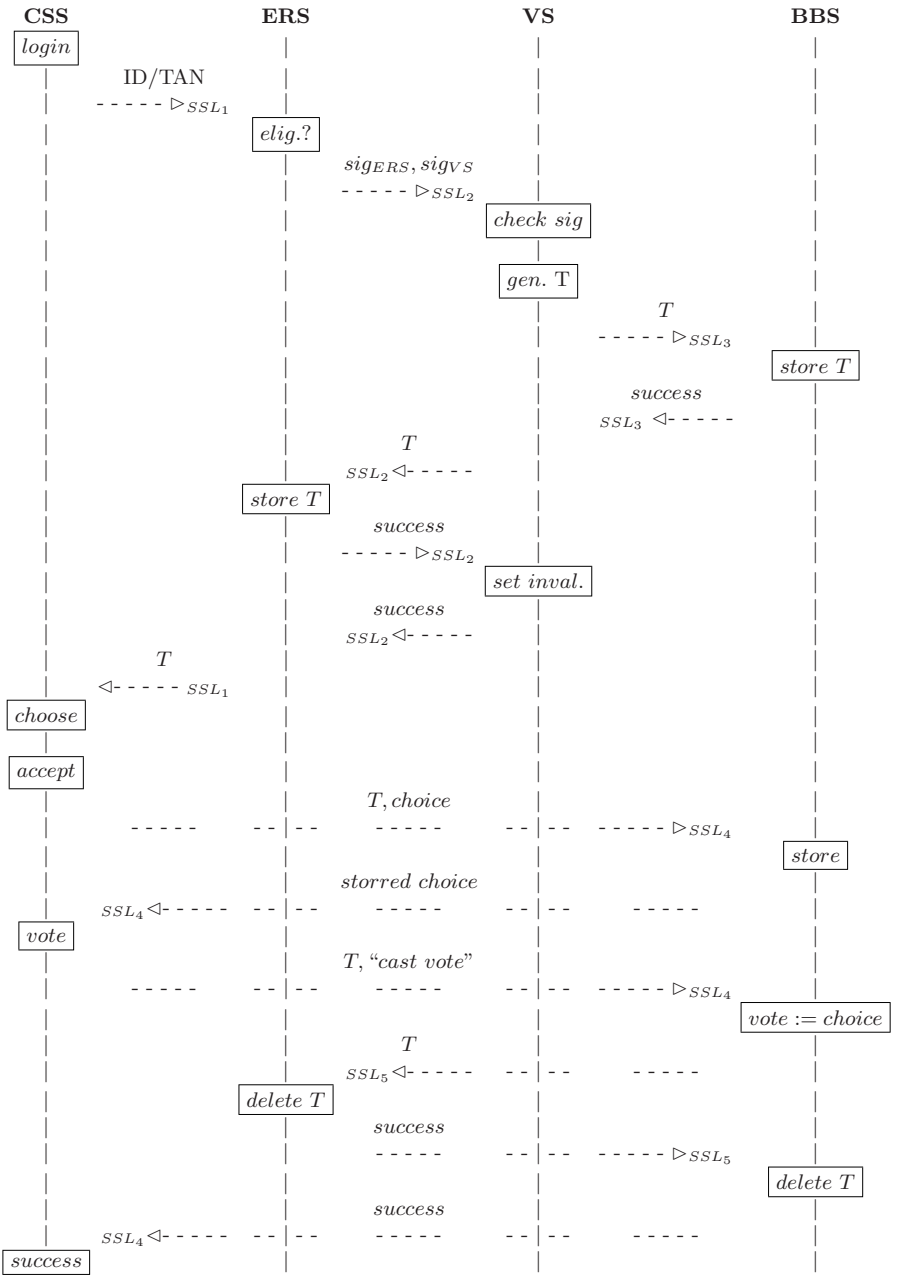

Fig. 9.3. The POLYAS voting protocol

the other two servers. Afterwards, the member of the responsible election authority needs to enter the pass phrase for the secret database key of the BBS. Thus, the e-votes can be decrypted, and the tallying software can calculate the election result.

9.3.2 System Analysis

A related analysis has been done by the developer in [123]. Here, the POLYAS system is analysed according to an older version of the GI/BSI/DFKI Protection Profile. The adoption of the analysis result from [123] and version 0.27 of the GI/BSI/DFKI Protection Profile is presented in Table 9.3 and 9.4. Several times, results from [123] are cited (together with a label from the corresponding security objective in this paper).

Result. There are some problems in deciding upon PASS or FAIL because the POLYAS system only provides a responsible election authority interface to start the election, while to stop the polling phase and to start the tallying shell, commands are used. The quantitative result shows that almost all security objectives deduced from threats got a PASS (10 of 12), while there is only one FAIL and one ORG. With respect to the security objectives related to organisational security policies, 7 security objectives are evaluated to PASS, 4 to ORG, 1 to FAIL, and 2 to INCONCL. The security objectives that are evaluated to FAIL are O.T.ElecSecrecyNet and O.OSP.Auditing. For all identified problems, [123] claims to have a solution. Thus, minor changes are necessary, in order to certify the POLYAS system, while these modifications are not related to the architecture or the voting protocol steps.

9.4 Summary

This chapter applies the developed evaluation framework (which is based on improvements to the GI/BSI/DFKI framework) to the two available systems: the Estonian system and the POLYAS system. As the analysis is different from a complete Common Criteria evaluation, Sect. 9.1 describes the applied evaluation procedure. This procedure is mainly based on the security problem definition. Moreover, the analysis for both systems starts with a detailed system description.

Section 9.2 addresses the Estonian system and Sect. 9.3 examines the POLYAS system. Based on the security problem definition, both systems (at least in the analysed version) do not meet all the specified security objectives. However, only minor modifications are necessary for both systems to be compliant[9], while the improvements are not related to the voting protocol.

[9] This compliance makes only statements on the system providing enough security functions to meet the security objectives (which corresponds to a Security Target evaluation), while a Common Criteria evaluation according to EAL2+ contains much more security assurance requirements.

Table 9.3. Result of the analysis for the POLYAS system (part 1)

Security Objective	Result	Explanation
O.T.IneligVoter	PASS	"It is only possible to get a voting token enabling a voter to cast a vote after sending the ID and the voting TAN to the ERS, which decides whether the request comes from an eligible voter who has not yet cast a vote. Without having such a valid voting token you can send vote messages to the BBS but these are rejected." (SecObj1)
O.T.OneVoterOneVote	PASS	See O.T.VoteRight (SecObj26)
O.T.ProofGen	PASS	According to Fig. 9.3, it is not possible to generate a proof from any information either sent to, displayed on, and/or sent from a vote-casting device. (SecObj2)
O.T.DeleteMsgNet	PASS	According to Fig. 9.3, this is ensured by SSL as long as the voter verifies the server certificate and ensures that he receives the last confirmation. (SecObj3)
O.T.AlterMsgNet	PASS	See O.T.DeleteMsgNet. (SecObj3)
O.T.ElectionSecrecyNet	FAIL	"First of all, the vote is transmitted encrypted via SSL. Secondly, the vote is not sent together with the identification data, not even during one SSL session. Thus, one can only link the encrypted identification data to the encrypted vote via corresponding sender IP addresses. The current problem is that someone who is observing the Internet and knows, which IP-address a particular voter has, can limit the possible choices the voter makes because of the size of the vote message. Especially, casting an invalid vote by choosing all candidates is observable." (SecObj4)
O.T.IntResultNet	PASS	See O.T.DeleteMsgNet. (SecObj5)
O.T.WrongServer	PASS	As SSL is used, this security objective is ensured as long as the voter verifies the voting server's SSL certificate.
O.T.IntegElecData	ORG	"After the completion of the result computation, POLYAS computes a hash value of the electoral register (including those who cast a vote and who did not) and a hash value of all votes. These two hash values are printed immediately and are part of the election commission documentation, which is signed by the election commission." (SecObj6)
O.T.ElectionSecrecy	PASS	"The only link between a voter and his vote on the server-side is the voting token. But the voting token is deleted at the ERS and the BBS just after completing the voting process for the corresponding voter. Thus, even knowing all data from the servers after the election it is not possible to compromise the secrecy of the vote because the link was already removed during the election." (SecObj7)
O.T.PersonalDataNet*	PASS	The identification data (ID) sent in the first steps of the protocol is secured with SSL. (SecObj5)
O.T.SecretAuthNet*	PASS	See O.T.DeleteMsgNet.

Table 9.4. Result of the analysis for the POLYAS system (part 2)

Security Objective	Result	Explanation
O.OSP.Interface	PASS	All required functionality is implemented. (SecObj8/19/20/22)
O.OSP.PWClosePoll	ORG	"At the particular day and time the election commission meets in order to first deactivate the VS and the BBS and later the ERS. But it is not controlled by POLYAS whether the end of the election is already reached." (SecObj9)
O.OSP.PWInterface	PASS ORG	"there is no functionality implemented for the election commission to access the (encrypted) votes [...]" (SecObj11), "[...] to access the database containing the (encrypted) votes (other than for the result computation) [...]" (SecObj12), and "[...] to access the electoral register [...]" (SecObj14/15/17). However, the reset functionality (SecObj13/16) and, thus, the calculation of intermediate results (SecObj18/30) is only ensured by organisational means.
O.OSP.Confirmation	PASS	See Fig. 9.3. (SecObj21)
O.OSP.SelfCheck	ORG	"Before the election each part of the software is digitally signed, meaning at any time the two election commission members responsible for a particular server can access the server and check whether the software running is still the one that has been installed. Moreover, the servers are observed using the Nagios software. This software checks regularly whether the server and the databases are still online and available."(SecObj23)
O.OSP.ErrorRecovery	ORG	"A comprehensive and exhaustive recovery concept has been developed containing all possible breakdown and restart scenarios. In case of system breakdowns, including data loss the election commission is informed and possible actions are discussed (is a restart possible?)."(SecObj24); also (SecObj28)
O.OSP.Auditing	FAIL	"Most of the events listed above are logged by POLYAS. The election data stored at the beginning of the election and the results after the counting process are missing in the current version. The audit records can be read on the corresponding server." (SecObj25)
O.OSP.VoteRight	PASS	"The POLYAS software installed on the ERS ensures that only those voters having valid IDs and voting TAN can continue the voting process and then cast a vote. It also ensures that all such voters can continue the voting process." (SecObj26)

Table 9.4 (continued)

Security Objective	Result	Explanation
O.OSP.VoteRightExc	PASS	"[...] the one voter-one vote principle can be ensured for all these situations as long as the voter takes care that in the event of not having received the final receipt, he or she needs to re-login to complete the voting process." (SecObj27)
O.OSP.SepDuty	PASS	Two administrators need to enter their passwords.
O.OSP.AC	ORG	The voting server's access control is used. (SecObj29)
O.OSP.AccurCalc	PASS	"The source code has been examined by the Physikalisch-Teschnische Bundesanstalt (PTB). They especially checked the vote casting algorithm."(SecObj31)
O.OSP.Feedback*	INCONCL	–
O.OSP.NoInteract*	INCONCL	–

Making the required modifications would mean that both systems could get certified in general.

As both systems are based on different architectures, different authentication techniques, different approaches to ensure the secrecy of the vote, and different implementations for the client-side voting software, it can be concluded that the proposed evaluation framework is very flexible. Moreover, no improvements for the framework can be deduced from this analysis.