Extending the Helios Internet Voting Scheme Towards New Election Settings

Vom Fachbereich Informatik der

Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades

Doktor rerum naturalium (Dr. rer. nat.)

von

Dipl.-Math. Oksana Kulyk

geboren in Tschernihiw, Ukraine.

Referenten:	Prof. Dr. Melanie Volkamer
	Prof. Dr. Marc Fischlin
Tag der Einreichung:	01.03.2017
Tag der mündlichen Prüfung:	24.04.2017
Hochschulkennziffer:	D17

Darmstadt 2017

List of Publications

E-Voting (Used in Thesis)

[1] Oksana Kulyk and Melanie Volkamer. A proxy voting scheme ensuring participation privacy and receipt-freeness. In *E-Vote-ID 2017: 2nd International Joint Conference on Electronic Voting*. IFCA, April 2017. Submitted, under review.

[2] Oksana Kulyk, Karola Marky, Stephan Neumann, and Melanie Volkamer. Enabling vote delegation in boardroom voting. In *FC 2017: 2nd Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2017*. IFCA, April 2017. In press.

[3] Oksana Kulyk, Karola Marky, Stephan Neumann, and Melanie Volkamer. Introducing proxy voting to Helios. In *ARES 2016: 11th International Conference on Availability, Reliability and Security*, pages 98–106. IEEE, September 2016.

[4] David Bernhard, Oksana Kulyk, and Melanie Volkamer. Security proofs for participation privacy and verifiability for Helios. Cryptology ePrint Archive, Report 2016/431, May 2016. Submitted to *ARES 2017: 12th International Conference on Availability, Reliability and Security*, under review.

[5] Oksana Kulyk, Vanessa Teague, and Melanie Volkamer. Extending Helios towards private eligibility verifiability. In *VoteID 2015: E-Voting and Identity, 5th International Conference*, pages 57–73. Springer, September 2015.

[6] Oksana Kulyk, Stephan Neumann, Christian Feier, Melanie Volkamer, and Thorben Köster. Electronic voting with fully distributed trust and maximized flexibility regarding ballot design. In *EVOTE 2014: 6th International Conference on Electronic Voting*, pages 1–10. IEEE, October 2014.

E-Voting (Not Used in Thesis)

[1] Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, and Melanie Volkamer. Coercion-resistant proxy voting (extended version). *Computers & Security*, 2017. Revision submitted, under review.

[2] Karola Marky, Oksana Kulyk, Stephan Neumann, and Melanie Volkamer. Analysis of the proxy voting implementation LiquidFeedback. In *Journal of Information Security and Applications*, 2017. Submitted, under review.

[3] Oksana Kulyk, Stephan Neumann, Jurlind Budurushi, and Melanie Volkamer. Nothing comes for free: How much usability can you sacrifice for security? *IEEE Security & Privacy Special Issue on Electronic Voting*, 2017. In press.

[4] Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, and Melanie Volkamer. Coercion-resistant proxy voting. In *IFIP SEC 2016: 31st International Conference on ICT Systems Security and Privacy Protection*, pages 3–16. Springer, June 2016.

[5] Oksana Kulyk and Melanie Volkamer. Efficiency comparison of various approaches in e-voting protocols. In *FC 2016: 1nd Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2016*. Springer, February 2016.

[6] Oksana Kulyk, Stephan Neumann, Jurlind Budurushi, Melanie Volkamer, Rolf Haenni, Reto Koenig, and Philemon von Bergen. Efficiency evaluation of cryptographic protocols for boardroom voting. In *ARES 2015: 10th International Conference on Availability, Reliability and Security*, pages 224–229. IEEE, August 2015. An extended version of this work is available under http://eprint.iacr.org/2015/558.

[7] Stephan Neumann, Oksana Kulyk, and Melanie Volkamer. A usable Android application implementing distributed cryptography for election authorities. In *FARES 2014: 9th International Workshop on Frontiers in Availability, Reliability and Security*, pages 198–207. IEEE, September 2014.

Usable Security (Not Used in Thesis)

[1] Oksana Kulyk, Benjamin Maximilian Reinheimer, Paul Gerber, Florian Volk, Melanie Volkamer, and Max Mühlhäuser. Advancing trust visualisations for wider applicability and user acceptance. In *TrustCom 2017: 16th IEEE International Conference On Trust, Security And Privacy In Computing And Communications*, 2017. Accepted for publication.

[2] Oksana Kulyk, Benjamin Reinheimer, and Melanie Volkamer. Sharing information with web services – a mental model approach in the context of optional information.

In *HCII 2017: 19th International Conference on Human-Computer Interaction*, July 2017. In press.

[3] Oksana Kulyk, Paul Gerber, Michael El Hanafi, Benjamin Reinheimer, Karen Renaud, and Melanie Volkamer. Encouraging privacy-aware smartphone app installation: What would the technically-adept do. In *USEC 2016: 4th Usable Security Workshop*. Internet Society, February 2016.

[4] Melanie Volkamer, Karen Renaud, Oksana Kulyk, and Sinem Emeröz. A socio-technical investigation into smartphone security. In *STM 2015: 11th International Workshop on Security and Trust Management*, pages 265–273. Springer, September 2015.

Acknowledgements

I am grateful to my supervisor Prof. Dr. Melanie Volkamer for all her hard work and patience, as well as for her support, inspiration and opportunities that she has provided me during the course of my work.

I also thank Prof. Dr. Marc Fischlin for agreeing to be my coreferent and for providing me with helpful feedback.

I am honored to have Prof. Dr. Stefan Katzenbeisser, Prof. Dr. Reiner Hähnle and Prof. Dr. Michael Waidner on my PhD committee.

I thank my colleagues in the SECUSO research group for their helpful comments and discussions, especially Stephan Neumann, Peter Mayer, Jurlind Budurushi, Karola Marky and Birgit Henhapl.

I am also grateful to all my coauthors I thank Vanessa Teague, Rolf Haenni, Reto Koenig, Karen Renaud, David Bernhard, Tomasz Truderung for providing me with feedback regarding writing and proof reading my thesis. I would further like to thank the other members of electronic voting community for their support, comments and fruitful discussions.

Thanks to the services of Mentorium GmbH and of Robert Kinsella from Language Boutique in proof reading the language of my thesis.

I thank my parents by providing me their support during all my life and the time of my PhD in particular.

Wissenschaftlicher Werdegang

April 2006 – August 2012 Studium der Mathematik an der Justus-Liebig Universität Gießen mit Nebenfach Informatik. Abschluss: Diplom

Februar 2013 – Oktober 2013 Wissenschaftliche Hilfskraft in der SECUSO ("Security, Usability, Society") Forschungsgruppe unter der Leitung von Prof. Dr. Melanie Volkamer der Technischen Universität Darmstadt

seit Oktober 2013 Wissenschaftliche Mitarbeiterinin der SECUSO ("Security, Usability, Society") Forschungsgruppe unter der Leitung von Prof. Dr. Melanie Volkamer der Technischen Universität Darmstadt

Abstract

Internet voting has long been a topic both of public discussion and also of scientific research. While the introduction of Internet voting may bring many advantages, it is further important to ensure an adequate level of security of the systems and underlying schemes that are used for casting and tallying the votes in order to encourage faith and acceptance for this relatively new way of voting.

A number of cryptographic schemes have been proposed, that enable secure Internet voting. One of the most established and well-researched solutions is the Helios scheme, which is also implemented as an open-source system. Both its implementation and the scheme behind it has been extensively studied in the literature, and the Helios system has been used for numerous elections in practice, such as the IACR elections [IAC16].

However, there are election settings for which Helios is currently not appropriate, either due to infrastructure demands, required functionality for the voters or assurance of the security requirements. These kinds of election settings could benefit from the advantages that secure Internet voting provides.

In this thesis we identify the election settings not currently supported by Helios, propose our extensions for each one of these settings and evaluate their security. Hence, this work describes four Internet voting schemes that are build upon Helios, with each scheme developed towards a specific setting.

The first scheme presented here enables elections within the so-called boardroom voting setting. This setting is characterized by its decentralization, whereby all the tasks within the election are distributively performed by the voters themselves, without the support of a centralized infrastructure. The election in the boardroom voting setting are further conducted in an ad-hoc manner, so that limited time is available for preparation beforehand. We propose an extension of Helios that distributes the tasks of the voting system components in Helios among the voters. For this, we use cryptographic primitives such as decentralized key exchange with short authentication strings, distributed threshold secret sharing and distributed threshold decryption and Byzantine agreement.

The second scheme extends Helios with proxy voting functionality. Proxy voting, as a newly emerged form of voting, enables the voter to delegate her voting right in the election to a trusted third-party, the so-called proxy, who is authorized to vote on the voter's behalf. This extension facilitates such delegation while assuring the security for

delegating voters and for the proxies and preserves the security guarantees provided by Helios for the voters who vote directly (instead of delegating). For ensuring the security of our extension, we introduce the so-called delegation credentials that are assigned to the voters and are used to compute anonymized delegation tokens sent to the proxies to enable delegation. We further use cryptographic primitives such as proofs of knowledge and signatures of knowledge.

The third scheme combines the first two settings to extend Helios towards the proxy boardroom voting setting, namely, a setting in which the elections are performed in a decentralized way as in boardroom voting, yet the voters who cannot participate in the election themselves are allowed to delegate their voting right to a trusted proxy before the election. The security of our extension is assured with threshold secret sharing and Pedersen commitments.

The fourth scheme extends Helios by improving its security. As such, it introduces participation privacy, meaning that the voting system does not reveal which voters have participated in the election, while supporting verification that only the eligible voters have cast their ballots in the election. The extension furthermore introduces receipt-freeness, ensuring that the voter cannot create a receipt that proves to a third party how she voted, thus preventing vote selling. To ensure the security of the extension, a new kind of entity is introduced, the posting trustee, and a new kind of ballot, the so-called dummy ballot that is indistinguishable from a normal ballot cast by the voter, but does not modify the election result. We furthermore use disjunctive zero-knowledge proofs and proofs of signature knowledge to prove, that a sender of a particular ballot knows the private signature key of an eligible voter, or that the ballot is a dummy ballot.

For each one of the extensions, the security model is provided, which describes the security requirements and the assumptions that are necessary for ensuring the security requirements (i.e. vote privacy or vote integrity), is provided. For the first three extensions, the security model is used as a base for the informal security evaluation, in which an informal argument is used to show, that the security requirements hold under the described assumptions. Conducting a formal security evaluation for these extensions is considered an important part of the future work, in which new formal definitions have to be developed. For the fourth extension, we provide a formal security analysis that relies on the formal definitions for the security requirements of vote privacy, vote integrity and eligibility, available in the literature. We furthermore introduce new formal definitions for participation privacy, receipt-freeness and fairness, which we also use for the formal proofs of our extension.

Zusammenfassung

Internetwahlen sind bereits seit Langem ein Thema in der öffentlichen Diskussion sowie in der wissenschaftlichen Forschung. Während die Einführung von Internetwahlen viele Vorteile mit sich bringen kann, ist es ebenso wichtig ein adäquates Niveau von Sicherheit zu garantieren, sodass Vertrauen und Akzeptanz in diese relativ neue Wahlform aufgebaut werden kann. Die Sicherheit bezieht sich speziell auf Systeme und die dahinter liegenden Protokolle, die für die Stimmabgabe und die Auszählung benutzt werden.

Es wurde bereits eine Vielzahl kryptographischer Protokolle vorgeschlagen, die sichere Internetwahlen ermöglichen sollen. Darunter ist Helios das am meisten etabilierte und erforschte Protokoll, für das es zusätzlich auch eine Open-Source Implementierung gibt. Sowohl diese Implementierung als auch das Protokoll dahinter wurden extensiv in verschiedener Literatur studiert, und das Helios Wahlsystem wurde für mehrere Wahlen in der Praxis benutzt, darunter die Wahlen der IACR [IAC16].

Jedoch gibt es einige Wahlszenarien, für die Helios momentan noch nicht geeignet ist. Darunter fallen Infrastrukturanforderungen, vom Wähler benötigte Funktionalität oder Sicherheitsanforderungen. Dennoch können diese Arten von Wahlen von den Vorteilen, die sichere Internetwahlen mit sich bringen, profitieren.

In dieser Dissertation identifizieren wir Wahlszenarien, die momentan nicht vom Helios unterstützt werden, schlagen Erweiterungen für jedes dieser Szenarien vor und evaluieren die Sicherheit dieser Erweiterungen. Infolgedessen beschreibt diese Arbeit vier Internetwahlprotokolle die auf Helios aufbauen, wobei jedes Protokoll für ein spezifisches Wahlszenario entwickelt wurde.

Das erste Protokoll, das hier präsentiert wird, ermöglicht sogenannte Wahlen in Gremien ("Boardroom Voting"). Dieses Szenario ist durch seine Dezentralisierung charakterisiert, wobei alle Aufgaben während einer Wahl auf die Wählern selbst, ohne Unterstützung durch eine zentralisierte Infrastruktur, verteilt sind. Die Wahlen in Gremien werden ad-hoc durchgeführt, sodass wenig Zeit für Vorbereitungen im Voraus zur Verfügung steht. Wir schlagen eine Erweiterung von Helios vor, die die Aufgaben der Wahlsystemkomponenten in Helios auf die Wähler verteilt. Dafür benutzen wir kryprographische Primitive wie den dezentralisierten Schlüsselaustausch mit kurzen Authentifizierungszeichenketten, verteiltes Secret-Sharing, verteilte Entschlüsselung und die Byzantinische-Fehler-Toleranz.

Das zweite Protokoll erweitert Helios mit der Funktionalität für Wahlen mit Delegation-

sprinzip ("Proxy Voting"). Proxy Voting ist eine Wahlform, die es dem Wähler erlaubt sein Wahlrecht in einer Wahl an eine vertrauenswürdige Drittpartei, dem sogenannten Proxy, zu delegieren. So wird der Proxy bevollmächtigt für diesen Wähler eine Stimme abzugeben. Diese Erweiterung ermöglicht das Delegieren der Stimme und gewährleistet dieselben Sicherheitseigenschaften für nicht-delegierende Wähler wie im Original-Helios. Um die Sicherheit unserer Erweiterung zu garantieren, führen wir sogenannte Delegierungs-Credentials ein. Diese sind den Wählern zugeordnet und werden verwendet, um die anonymisierte Delegierungs-Tokens zu erzeugen. Der Wähler sendet zum Delegieren ein Delegierungs-Token an einen Proxy. Ferner benutzen wir kryptographische Primitive wie Zero-Knowledge Beweise und Signatures-of-Knowledge.

Das dritte Protokoll kombiniert die ersten beiden Szenarien, um Helios für Wahlen mit Delegationsprinzip in Gremien ("Proxy Boardroom Voting") zu erweitern. In diesem Szenario werden die Wahlen in dezentralisierter Weise durchgeführt, dennoch haben die Wähler, die selbst nicht am Treffen teilnehmen können, die Möglichkeit ihr Wahlrecht an einen vertrauenswürdigen Proxy vor der Wahl zu delegieren. Die Sicherheit von unserer Erweiterung wird durch das Threshold-Secret-Sharing und Pedersen Commitments gewährleistet.

Das vierte Protokoll verbessert die Sicherheit von Helios. Unsere Erweiterung führt die Anonymität ein, sodass das Wahlsystem nicht preisgibt, welche Wähler an der Wahl teilgenommen haben. Die Erweiterung erlaubt es jedoch zu verifizieren, dass nur wahlberechtigte Wähler gewählt haben. Weiterhin führt die Erweiterung die Quittungsfreiheit ein: Der Wähler kann keine Quittung erzeugen, um einer Drittpartei zu beweisen, wie er gewählt hat. Dadurch wird der Verkauf von Stimmen verhindert. Um die Sicherheit unserer Erweiterung zu gewährleisten, wird eine neue Entität, ein Posting Trustee, sowie eine neue Art von Stimmzetteln, ein sogenannter Dummy-Stimmzettel, eingeführt. Der Dummy-Stimmzettel ist von einem normalen Stimmzettel, den ein Wähler abgibt, nicht unterscheidbar. Außerdem hat er keine Auswirkung auf das Wahlergebnis. Um für einen Stimmzettel zu zeigen, dass entweder der Absender des Stimmzettels einen geheimen Signaturschlüssel des wahlberechtigten Wählers kennt oder dass der Stimmzettel ein Dummy-Stimmzettel ist, benutzen wir disjunktive Zero-Knowledge Beweise und Proofs-of-Signature-Knowledge.

Für jede Erweiterung wird ein Sicherheitsmodell angegeben, das die Sicherheitsanforderungen und die Annahmen, unter denen diese Sicherheitsanforderungen (z. B. Wahlgeheimnis oder Stimmintegrität) erfüllt werden, beschreibt. Für die ersten drei Erweiterungen wird das Sicherheitsmodell als Basis für eine informelle Sicherheitsevaluierung verwendet, wobei ein informelles Argument benutzt wird, um zu zeigen, dass die Sicherheitsanforderungen unter den genannten Annahmen erfüllt werden. Eine formale Sicherheitsevaluierung von diesen Erweiterungen durchzuführen stellt einen wichtigen Aspekt zukünftiger Arbeit dar. Dafür müssen jedoch zunächst neue formale Definitionen entwickelt werden. Für die vierte Erweiterung geben wir eine formale Sicherheitsanalyse an,

die auf formalen Definitionen für die Sicherheitsanforderungen von Wahlgeheimnis, Stimmintegrität und Wahlberechtigung auf vorhandener Literatur aufbaut. Weiterhin führen wir neue formale Definitionen für die Anonymität, Quittungsfreiheit und Fairness ein, die wir auch für die formalen Sicherheitsbeweise unserer Erweiterung benutzen.

Contents

Chapter 1

Introduction

1.1 Motivation

For many years, Internet voting has been a matter of public interest, both within the scientific community and as a subject of political debates. Its proponents stress the advantages of Internet voting, potentially increasing voter turnout and supporting voters who would otherwise experience difficulties casting their vote, such as voters abroad or for the voters with disabilities that impact their mobility. As a result, several countries, such as Estonia [Est10] and Switzerland [SGM+15], introduced Internet voting for legally-binding elections. Internet voting has been widely used on a smaller scale, such as in university elections [ADMP+09, Pri17] or elections in associations [OKNV12, IAC16].

However, it must be acknowledged that there is also strong opposition to the deployment of Internet voting. In particular, the opponents of Internet voting stress its vulnerability to cyberattacks on the voting system components which may manipulate the election result or violate vote privacy on a larger scale than it is possible with traditional paper-based elections. As such, vulnerabilities have been shown to exist in systems used for Internet voting in practice [WWIH12, SFD+14, HT15], that could well have been used by the attacker to manipulate the election outcome [SFD+14, HT15] or to reveal how each individual voter voted [WWIH12, SFD+14]. In particular, these vulnerabilities have shown that stronger mechanisms for ensuring vote privacy and for detecting manipulations within the elections are needed.

As a result, cryptographic solutions for Internet voting schemes have been proposed that aim to ensure such security requirements as vote privacy, vote integrity or eligibility (i.e. ensuring that only eligible voters participate in the election), as well as other security requirements deemed relevant to a particular election setting. These solutions, in particular, rely on such techniques as encryption, proofs of knowledge, or mix nets.

One of the Internet voting schemes that uses cryptography for ensuring its security is the Helios scheme [Adi08], implemented as an open-source voting system. Helios has been widely used for conducting small-scale Internet voting elections. The example of such elections include the university election at UC Louvain [ADMP+09], the university elections

in Princeton [Pri17] and the annual internal elections at the International Association of Cryptographic Research (IACR) since 2010 [IAC16]. Several extensions of Helios have been proposed, introducing such modifications as distributing trust between several voting system components, using pseudonyms instead of voter identities for publishing the cast ballots (both proposed and implemented in Helios 2.0 described in [ADMP+09]), or using digital signatures for authenticating the voters instead of passwords [CGGI14]. A number of papers exposed the vulnerabilities of either the original Helios scheme or its extensions [BPW12,KTV12] and the implemented software [BHPS16,CFE16,GKV+16], also addressing the solutions on fixing these vulnerabilities. Furthermore, methods of provable security have been used to evaluate the security of either the original Helios scheme or its extensions, resulting in formal proofs for its security requirements, namely, vote privacy in [BCG+15] and vote integrity and eligibility in [CGGI14].

The Helios scheme, with its different extensions is versatile enough to adapt to various election settings. However, there are still some limitations. Among these are the requirements on the infrastructure used for the election, available functionality for the voters and the level of privacy or integrity that Helios offers. This thesis focuses on several the election settings, for which the current extensions of Helios would not currently be appropriate. The goal of this work is to extend Helios towards some of these election settings, hence providing ways to conduct secure Internet voting in them.

1.2 Contributions of the Thesis

This thesis describes the extensions of the Helios scheme towards different election settings. These settings encompass as different characteristics regarding the electorate and available infrastructure, as well as different functional and security requirements and are described below.

The security of the extensions proposed in this thesis is systematically evaluated. Thereby, the assumptions on adversarial capabilities are derived, that are required for the fulfillment of the security requirements in the extensions. For one of the extensions, formal security proofs are provided.

The structure of the thesis is summarized in Figure 1.1, with the arrows showing the dependencies between the chapters.

1.2.1 Boardroom Voting

The first election setting considered in this thesis places specific demands on the available infrastructure and is characterized by a specific electorate. The setting is the so-called “boardroom voting”. It encompasses the elections that occur within corporations, university governing bodies, and during various meetings. Elections and polls during meetings are difficult as decisions are required when those who vote are not physically present. So far, technology enables them to participate in public discussions (e.g., over video confer-

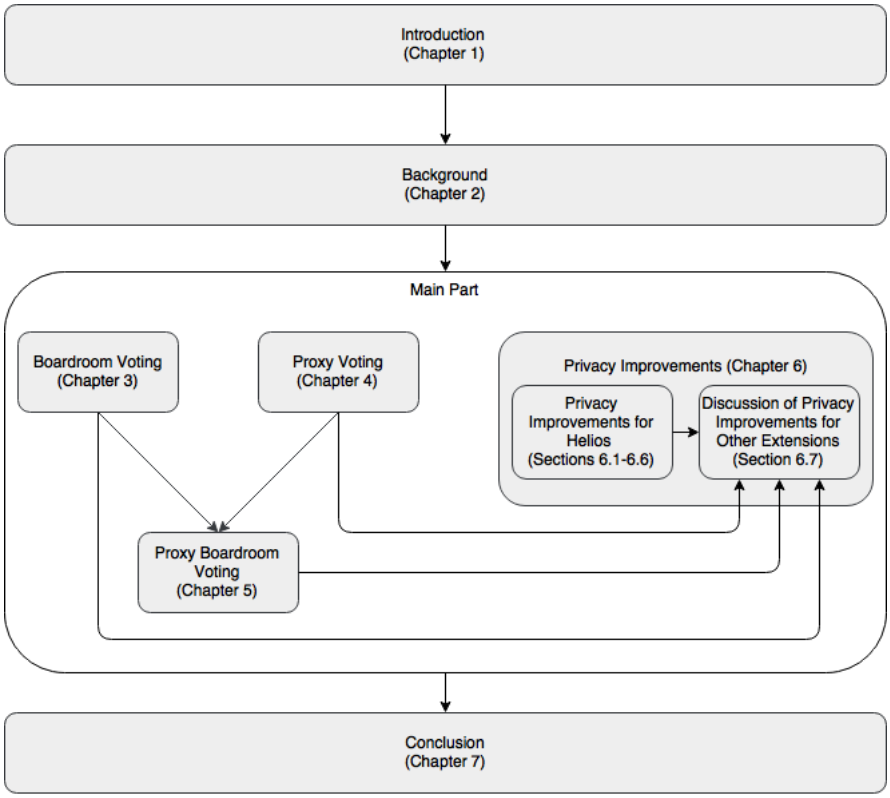


Figure 1.1: Structure of this thesis

ence), but then they are either excluded from the voting process, or the voting process is no longer secret if, for example, the voting is done via raising hands. Hence, a scheme for remote electronic voting would benefit such voters by allowing them to participate in the election while preserving vote privacy. In the boardroom voting setting, as opposed to large-scale elections, the voting is performed in smaller groups, often without specialized central election infrastructure, and is often conducted in an ad-hoc fashion. Hence, an Internet voting scheme that can be used for boardroom voting should enable decentralized and spontaneous elections.

There are extensions of Helios that offer some degree of decentralization by introducing multiple trustees responsible for tallying. However, they still depend on a central infrastructure such as a bulletin board or a registration authority to conduct elections.

Hence, the first contribution of this thesis is an extension of Helios which facilitates boardroom voting in order to support decentralized ad-hoc elections where some voters may not be co-present. For this extension we modify the tasks performed by the voting system components in Helios, so that they can be performed by the voters themselves in a distributed way.

The extension of Helios towards boardroom voting and its security evaluation is described in Chapter 3.

1.2.2 Proxy Voting

The second contribution of this thesis considers a form of voting, the so-called "proxy voting", where the voter has the right not only to cast the vote directly, but also to delegate it to a trusted person referred to as proxy. Such a form of voting can, for example, be useful in elections that occur on a frequent basis, so that the voters might easily become overwhelmed by the frequent demands to vote. Proxy voting would enable them to delegate some of these decisions, while (as opposed to representative democracy) they still retain the rights to vote directly on other issues, on which the voters feel more informed. As such, enabling proxy voting requires additional functionality that should be available to the voter, namely, being able to delegate her vote to a chosen proxy. Note that in this setting the voter does not provide the proxy with instructions on how to vote, since the purpose of the delegation is to support voters who are not sure how to vote but want to delegate their vote to a person whose decision making in choosing a voting option to vote for they trust.

Generally, the original Helios scheme as well as its existing extensions allow the voter to delegate her vote to a proxy. The ways to do this, however, have their disadvantages. As such, the first way the voters can delegate their vote in Helios is to divulge their voting credentials to a proxy, in effect allowing them to vote on her behalf. In this case, if the voter changes her mind, wanting to cancel her delegation and vote directly, there is no simple mechanism for her to do so without contacting the registration authority. Besides, this solution implies that the proxy knows which voter delegated to her, if the voter credentials are tied to their identities. A second way to delegate using Helios is for the proxies to prepare and submit her ballot privately to the voters who request it. This solution, however, also has disadvantages, as the proxy has to make her choice in advance, before the voters delegate to her, and cannot change her vote after the voters have cast her ballot without having to contact the voters again. She further knows the identities of the voters who delegated to her.

Hence, the second goal of this work is to extend Helios towards proxy voting while enabling the voters and the proxies to retain control over their votes, direct or delegated. As the security and functional requirements towards proxy voting have not been extensively studied in the literature, we first offer a list of security and functional requirements that we consider relevant for proxy voting. Then, we propose a scheme that, in addition to ensuring the security of original Helios for the voters who vote directly, addresses the proxy voting specific requirements thus securing the delegation process.

The extension of Helios towards proxy voting and its security evaluation is described in Chapter 4.

1.2.3 Proxy Boardroom Voting

The third contribution of this thesis considers the setting that requires proxy voting functionality in boardroom voting. In this setting we consider an election among boardroom

members, similar to boardroom voting that is performed in a decentralized way. The proxy voting functionality should ensure that the boardroom members who are unable to participate in the election (for example, by not being able to attend the meeting due to time constraints) are able to delegate their vote to a trusted proxy (either another boardroom member or a third party) who participates in the meeting and vote on behalf of the absent voter.

As such, the proposed setting suggests, that the functionality that enables to delegate one's vote *before* the meeting – hence, before the election starts – is required. The solution for proxy voting in centralized elections proposed in this thesis, on the other hand, implies that the voters delegate their votes after the election has been fully set up and the voting has started. Hence, a straightforward application of the extension towards proxy voting is not appropriate for proxy boardroom setting.

The extension towards boardroom voting would allow to delegate one's vote before the election in boardroom voting setting if the voter issues a signed form that enables the chosen proxy to cast a vote on behalf of this voter. However, such a solution reveals both to the proxy and to the rest of the voters would know to which proxy the voter has delegated her vote. This is justified in some cases, whereby other boardroom members know, whom the delegating voter trusts anyway. Still, in other cases the voter does not wish to publicly disclose her support for a particular proxy to others, or even to the proxy herself.

Hence, the third contribution of the thesis is extending Helios towards proxy boardroom voting by combining and modifying the ideas from the extensions for boardroom voting and proxy voting.

The extension towards proxy boardroom voting and its security analysis is described in Chapter 5.

1.2.4 Privacy Improvements

The final contribution considers an election setting that requires a higher level of privacy than Helios provides. Namely, it aims to ensure *participation privacy*, meaning that the information, whether or not a particular voter has participated in the election, should be hidden, and provide *receipt-freeness*, meaning that the voter should not be able to create a receipt that proves to a third party that she has voted for a particular candidate.

The original Helios scheme does not ensure participation privacy, as the identities of the voters who cast their ballot in the election are published on the bulletin board for the public to see. Subsequent extensions of Helios address this shortcoming by allowing the election organizers to assign pseudonyms to the voters [ADMP$^+$09], which are published instead of their identities. This solution, however, prevents from verifying the eligibility of the election, as the entity who assigns the pseudonyms should be trusted to issue them only to eligible voters.

As a further privacy issue, Helios does not ensure receipt-freeness. As such, if a voter

manages to use a modified version of a voter client, she can prove that the ballot she cast – which is published on the bulletin board next to her identity or pseudonym – was cast for a particular candidate. In this way, by forwarding this receipt to a third party, she can sell her vote. Note, as these receipts do not require extensive two-way communication or face-to-face meetings between the voter and the vote buyer, it is possible for the vote buying to occur on a large scale, potentially altering the outcome of the election.

Hence, the final contribution of this thesis is an extension of Helios towards privacy improvements that achieves two goals: ensuring participation privacy while still being able to publish the identities of the eligible voters instead of their pseudonyms in order to enable the verification of eligibility, and ensuring receipt-freeness. Additionally, the ways to introduce participation privacy and receipt-freeness into the extensions towards boardroom voting, proxy voting and proxy boardroom voting are discussed.

A formal security analysis of the proposed extension with privacy improvements is provided. It relies on the existing formal security definitions that were used in order to prove the fulfillment of the security requirements such as vote privacy, vote integrity and eligibility in Heliios. Furthermore, new formal definitions for participation privacy and receipt-freeness are introduced. These new definitions are used for proving the fulfillment of these requirements in the proposed extension.

The extension of Helios towards privacy improvements of participation privacy and receipt-freeness and its security analysis is described in Chapter 6. Futhermore, Chapter 6 discusses the ways to introduce the privacy improvements of participation privacy and receipt-freeness into the settings described in Chapters 3 to 5.

Chapter 2

Background

This chapter describes the background of the content of the thesis. First, the security requirements are provided that have been considered for Internet voting schemes. Then the cryptographic primitives are outlined that are used in the Internet voting schemes, in particular the schemes proposed in this thesis. Finally, an overview of Helios is given, including the description of the variant of Helios that is used as a basis for the extensions proposed in the thesis and a security model for it.

The contents of this section have been partially published at the *11th International Conference on Availability, Reliability and Security* [KMNV16], at the *6th International Conference on Electronic Voting, Verifying the Vote* [KNV+14] and at the *5th International Conference on E-Voting and Identity* [KTV15].

2.1 Security Requirements

In this section we describe the security requirements for Internet voting used in this work.

Security requirements in Internet voting have been an extensive research topic, both from the technical and from the legal perspective. In particular, the legal perspective has been considered in [Vol09, BGRR13, LSBV10, Neu16] by deriving the requirements from the legal election principles. A set of recommendations has been proposed by the Council of Europe in 2004 [Cou04]. From the technical perspective, a number of definitions of the security requirements have been used for the specification and evaluation of the Internet voting schemes, both formally (e.g. [BCG+15, DKR09, CGKü+16, SFC15]) and informally. These requirements are summarized and described below.[1]. Note, that these requirements can be ensured only for the voters who are not completely controlled by the adversary.

As such, several of these requirements relate to the privacy aspect of Internet voting security, ensuring that no information about the voter's intention is leaked, aside from what can be deduced from the election result. Thereby one can distinguish between following aspects of privacy, depending on the data that should be private:

[1] Since there is no consistent terminology in the literature for some of the requirements, we list the requirements by the terms that we are going to use in this work.

Vote Privacy. The voting system should not provide additional information on how each particular voter has voted [RBH+09, KR05, DKR09], aside from the information available from the election result.

Fairness. The voting system should not reveal any partial results before the voting is finished [KR05].

Participation Privacy. The voting system should not reveal whether a particular voter has participated in the election[2] [HS11, CGGI14].

The literature further defines such security properties as *receipt-freeness*, *coercion-resistance* or *everlasting privacy*, that aim to preserve the vote privacy or the participation privacy requirements under specific assumptions on the security model. As such, receipt-freeness means that the vote privacy requirement should be preserved even for the voters who attempt to obtain a receipt that can prove to a third party how they voted [RBH+09, KR05, DKR09]. An even stronger property is coercion resistance, meaning that the vote privacy and participation privacy are preserved even in case of voter coercion, i.e. in case the voter chooses to cooperate with the adversary during voting [RBH+09, DKR09, JCJ05]. Another privacy-related property mentioned in the literature is everlasting privacy, meaning that vote privacy should be ensured even for a computationally unrestricted adversary [MN06].

Other requirements are related to the integrity aspect of the Internet voting and are meant to ensure that the election has not been manipulated. Thus, the following steps of preventing the manipulations can be distinguished:

Vote Integrity. It should be ensured that all the votes are correctly processed by the voting system, without being altered or manipulated [RBH+09]. Often, three steps of processing the ballots are considered, distinguishing between *cast-as-intended* (i.e. the cast ballot corresponds to the intention of the voter), *stored-as-cast* (i.e. the cast ballot was stored correctly by the voting system) and *tallied-as-stored* (i.e. all the stored ballots are processed correctly during the tallying) vote integrity. [RBH+09, SFC15].

Eligibility. It should be ensured, that only the votes from the eligible voters, and only one vote from each voter, are included in the election result. [SFC15, LK02, KR05].

[2] While this requirement is most commonly referred to as "anonymity" in the literature, we propose the term "participation privacy" which we consider more accurate and more helpful in distinguishing this requirement from other privacy-related requirements.

Moreover, much of the focus in the Internet voting literature has been on providing verifiability for the elections, meaning that there should be means to detect manipulations by performing the required verifications without relying on a trusted entity.

Finally, for dealing with the reliability aspect of the Internet voting, a following requirement is defined to ensure that the election can be successfully conducted even in presence of some faulty system components or denial-of-service attacks:

Robustness. The voting system should be able to successfully complete the election after all the votes have been cast, even in case if some of the authorities fail to produce valid output [RBH+09, LK02].

In this work we consider the security requirements of *vote privacy*, *fairness*, *participation privacy*, *vote integrity*, *eligibility* and *robustness*.

2.2 Cryptographic Primitives

In this section we describe the cryptographic primitives used in both the original Helios scheme and in our extensions. We generally assume that the cryptographic problems, on which the security of described primitives depends, are computationally hard.

2.2.1 ElGamal

The ElGamal encryption scheme [ElG85] is a probabilistic public-key cryptosystem that relies on the Decisional Diffie-Hellman assumption. Let $\mathbb{G}_q$ as a cyclic group with prime multiplicative order q where the Decisional Diffie-Hellman is assumed to hold. Given a generator g of $\mathbb{G}_q$, the public key is defined as (g, h) with $x = \log_h g$ as a secret key. A message $m \in \mathbb{G}_q$ is encrypted as $\mathsf{Enc}(\mathsf{pk}, m) = (a, b) = (g^r, m \cdot h^r)$ for a randomly chosen $r \in \mathbb{Z}_q$. For decrypting the ciphertext (a, b) one computes $m = ba^{-x}$. The ElGamal encryption is either multiplicatively or additively homomorphic, with the latter property achieved by encrypting g^m instead of m. Note, that in case of an additively homomorphic ElGamal, only small values of m can be decrypted due to the complexity of calculating the discrete logarithm. Further in this thesis, $c_1 \cdot c_2 = (a_1, b_1) \cdot (a_2, b_2) = (a_1 \cdot a_2, b_1 \cdot b_2)$ denotes a pairwise multiplication of ciphertexts c_1, c_2.

2.2.2 Proofs of Knowledge

Proofs of knowledge are being commonly used for proving the knowledge of a witness for a particular statement. In particular, the proofs used in this thesis have the *honest verifier zero-knowledge* property, which means that the proof does not reveal any information about the witness in a communication with a honest verifier. Examples of such proofs

are the *proof of discrete logarithm knowledge* [Sch91], *proof of discrete logarithm equality* [CP92] or *knowledge of representation* [CS97a]. The following notation is used in this thesis. For example, given public parameters g, h and secret x, the proof of knowledge of the discrete logarithm x is denoted as:

$$\pi = PoK\{x : g^x = h\}$$

Cramer et al. [CDS94] propose the technique for the construction of *disjunctive witness-hiding* proofs that allow proving that the prover knows a witness to one out of multiple statements, without revealing the witness or the corresponding statement. Camenisch et al. furthermore describe the construction of proofs of knowledge for general statements of discrete logarithms has been proposed by Camenisch et al. in [CS97b].

For making proofs of knowledge used in this thesis into non-interactive zero-knowledge proofs of knowledge, thus making sure that no one learns anything from the proof aside from whether the prover knows a witness to the statement, the Fiat-Shamir heuristic [FS86] is used. Namely, the input from the verifier in a proof (the so-called *challenge*) is replaced with an output of a function $\mathcal{H}$, which should be indistinguishable from a random function. In practice, $\mathcal{H}$ is often instantiated as a cryptographic hash function. Depending on what $\mathcal{H}$ takes as an input, Bernhard et al. [BPW12] distinguish between weak and strong Fiat-Shamir heuristic: in the weak version, the input to $\mathcal{H}$ is only the first message from the prover (the so-called *commitment*), and in the strong version the input to $\mathcal{H}$ also includes the public values for the statement that is being proven.

2.2.3 Signatures of Knowledge

Proofs of knowledge can also be used as a digital signature scheme, in so-called *signatures of knowledge*, a concept, described in [CS97a]. Namely, given the message m to be signed, the Fiat-Shamir heuristic is used for computing the proof of knowledge so that m is included in the input for the function $\mathcal{H}$ that outputs the challenge in the proof. In this thesis, a notation similar to the proofs of knowledge is also used for the signatures of knowledge: for example, given (g, h) as a public key and $x = \log_g h$ as a corresponding secret key, a signature of knowledge on m, computed by proving the knowledge of x, is denoted as:

$$\pi = PoK\{x : g^x = h\}(m)$$

2.2.4 Proof of Encryption of 0

For proving that a given ElGamal ciphertext (a, b) encrypts 0 in additively homomorphic ElGamal (or 1 in multiplicatively homomorphic ElGamal), one presents the proof:

$$PoK\{\exists r : a = g^r p \wedge b = h^r\}$$

This is done using the proof of discrete logarithm equality described in [CP92].

2.2.5 Proof of Plaintext Knowledge

The ElGamal encryption does not provide non-malleability, meaning that given a ciphertext c one can calculate a ciphertext c' that encrypts a plaintext that is meaningfully related to the plaintext in c, without knowing the plaintext. As such, c' can be a re-encryption of c, so that two ciphertexts encrypt the same plaintext. While this property is useful in some cases, it is also that should be prevented in others, such as in order to protect from the ballot copying attacks in Internet voting [SB13] that could violate vote privacy. A simple way to introduce non-malleability to the ElGamal encryption, described in [BPW12], is to make the sender of the ciphertext prove that they know a corresponding plaintext. This can be done by using the non-interactive proof of knowledge of discrete logarithm (described in [Sch91]). Thus, for $c = (a, b) = (g^r, m \cdot h^r)$ with g, h being the ElGamal public keys, proving the knowledge of a plaintext m can be done via proving the knowledge of r given a. As shown in [BPW12], the ElGamal scheme with the proof of plaintext knowledge is NM-CPA secure.

2.2.6 Proof of 1-of-L Encryption

In order to prove that a ciphertext (a, b) encrypted with the ElGamal public key (g, h) encrypts a message m that is in a given finite set, i.e. that $m \in \{m_1, ..., m_L\}$, one computes the proof of knowledge with

$$\pi = PoK\{r : g^r = a \ \wedge \ \bigvee_{i=1}^{L} m_i h^r = b\}$$

For constructing this proof, the techniques for constructing the proof of knowledge for proving the equality of discrete logarithms [CP92] and for disjunctive proofs [CDS94] are being used.

2.2.7 Proof of Decryption Validity

The proofs of decryption validity are used in order to show that an ElGamal ciphertext (a, b) decrypts to a message m without revealing the private key used for decryption. Namely, given an ElGamal public key (g, h) and a message m, one proves that

$$\pi = PoK\{s : g^s = h \ \wedge \ a^s = b \cdot m^{-1}\}$$

The proof of decryption validity hence is equivalent for the proof of equality for discrete logarithms described in [CP92].

2.2.8 Proof of Signature Knowledge

Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a digital signature scheme, $\mathsf{pk}_s \leftarrow\!\!\$\, \mathsf{KeyGen}$ a public key for $\mathcal{S}$ and m a message from the message space of $\mathcal{S}$. The proof of signature knowledge is used for the prover to show that she knows a valid signature for m without revealing the signature itself, i. e.

$$\pi = PoK\{s : \mathsf{Verify}(\mathsf{pk}_s, m) = 1\}$$

The ways to construct a non-interactive proof of signature knowledge for some of the common digital signature schemes (e. g. RSA and DSA) are described in [ASW98].

2.2.9 Homomorphic Tallying

One of the approaches for anonymizing the ballots in Internet voting relies on the homomorphic properties of an ElGamal encryption system. In this way, one can compute a ciphertext that encrypts the sum of all the cast votes, so that the votes do not have to be decrypted individually in order to get the election result. Namely, it holds for the exponential ElGamal for the votes $v_1, ..., v_N$ and encryption function $\mathsf{Enc}(\mathsf{pk}, v)$:

$$\prod_{i=1}^{N} \mathsf{Enc}(\mathsf{pk}, v_i) = \mathsf{Enc}(\mathsf{pk}, \sum_{i=1}^{N} v_i)$$

The most common way is to encode the votes in such a way, that the voters cast either $v = 1$, which represents either a "yes"-vote or a vote in support of a specified voting option, or $v = 0$. Note, that in order prevent the manipulations of the election result while using the homomorphic tallying approach, it is important that the cast ballots represent the valid voting option. Otherwise, a malicious voter could manipulate the election result via either over-voting (i.e. casting a vote for $v = 100$) or negative voting (i.e. casting a vote $v = -1$). In order to prevent such manipulations, one proofs that the ciphertext cast with the ballot encrypts a valid voting option using the proof of 1-out-of-L encryption (Section 2.2.6).

2.2.10 Shamirs Secret Sharing

In order to enable sharing a secret between different parties, Shamir proposed a scheme for threshold secret sharing in [Sha79]. For sharing the secret m between N parties, so that at least t of them can reconstruct the secret together, the secret holder selects a polynomial $f(x) \in \mathbb{Z}_q[x]$ of degree $t-1$ with $f(0) = m$. A secret share m_i that is sent to a party $i = 1, ...N$ is calculated as $m_i = f(i)$. The reconstruction of the secret, given a set of at least t shares m_i, $Q \subset \{1, ..., N\}$, $|Q| \geq t$ is calculated as $m = \sum_{i \in Q} \lambda_i m_i$ with λ_i as a Lagrange coefficient $\lambda_i = \prod_{j \in Q, j \neq i} \frac{i}{i-j}$. Note that while the secret can be reconstructed given at least t shares m_i, a set of less than t shares m_i does not leak any information about the secret.

2.2.11 Distributed Threshold Secret Sharing and Distributed Threshold Decryption

In order to avoid having to trust a single entity that holds a secret, the secret sharing scheme of Shamir has been further extended by Pedersen [Ped91, Ped92a]. The scheme proposed by Pedersen enables to generate and share the secret in a decentralized manner among multiple parties, while enabling the parties to verify the correctness of their secret shares.

The Pedersen scheme has been used (as described in [CGGI13]) to generate a public key and share a corresponding private key for the ElGamal cryptosystem and distributively decrypt the ciphertexts that are encrypted using the generated public key. Cortier et al. further prove in [CGGI13], that the resulting cryptosystem is IND-CPA secure.

2.2.12 Verifiable Re-encryption Mix Net Schemes

In order to anonymize a list of ciphertexts, the mix net shuffle schemes have been developed. In particular, the re-encryption mix net schemes rely on the homomorphic property of an underlying cryptosystem. A number of entities, called the *mix nodes*, participate in the scheme, whereby each mix node in turn shuffles the list of encrypted ciphertexts $C = (c_1 = \mathsf{Enc}(\mathsf{pk}, m_1), ..., c_N = \mathsf{Enc}(\mathsf{pk}, m_N))$ using a secret permutation π and secret randomness values $r = (r_1, ..., r_N)$, outputting the shuffled list $C' = (c'_1, ..., c'_N)$ so that holds:

$$c'_i = \mathsf{Enc}_{\mathsf{pk},1} \cdot c_{\pi(i)}$$

In order to ensure that no ciphertexts have been manipulated during the shuffle, however, each node has to prove that the input and output set contain the same messages $m_1, ..., m_N$ (without revealing π and r). Hence, a number of proofs of shuffle validity have been developed [JJ00, DJV12, Gro10, BG12, TW10]. The comparison of these proofs is provided in 2.1, with N denoting the total number of ciphertexts and T as the total number of mix nodes. Thereby, $|A|$ denotes the size of an anonymity set, so that for a particular input ciphertext c and a subset of output ciphertexts A it is known, that the re-encryption of c is in A. The soundness is measured as a probability p, with which an adversary can provide a valid proof for a manipulated shuffle output. For measuring the efficiency of the proof, E denotes the number of modular exponentiations needed for computing the proof and the verification, and for measuring its robustness, t denotes the minimal number of the mix nodes required to successfully complete the shuffle.

The verifiable re-encryption mix net can also be used to shuffle tuples of ciphertexts $(c_{1,1}, ..., c_{1,k}), ..., (c_{N,1}, ..., c_{N,k})$, so that the order of the ciphertexts within a tuple $(c_{i,1}, ..., c_{i,k})$ is preserved. In that case, the proofs of shuffle validity such as [TW10, BG12] can be modified to prove, that the same permutation has been used for shuffling each vector $(c_{1,j}, ..., c_{N,j})$, $j = 1, ..., k$.

PoS	$\|A\|$	E	p	t
[JJ00]	$N/2$	$2N$	50%	$(t/2+1)$
[DJV12]	N	$6\sqrt{N}$	$(\sqrt{N}-1)/N$	1
[Gro10]	N	$12N$	*negligible*	1
[BG12]	N	$2N \log k + 4N$	*negligible*	1
[TW10, Wik09]	N	$19N + 15$	*negligible*	1

k is a divisor of N

Table 2.1: Comparison of mix net schemes

2.2.13 Pedersen Commitment

In order to commit to a value without revealing it, several commitment schemes have been developed. One of them, which we use in our extensions, is the Pedersen commitment [Ped92b], that is calculated as follows: Given two independent generators $(g, h) \in \mathbb{G}_q^2$, a commitment on a value $m \in \mathbb{Z}_q$ is calculated as $c = g^m h^r$ for a random value $r \in \mathbb{Z}_q$. The commitment reveals no information about m, so that even the computationally unrestricted adversary is unable to determine m given c. Furthermore, without knowing the discrete logarithm $\log_g h$, it is infeasible to find two different decommitment values m', m for c.

2.2.14 Public-Key Infrastructure.

A public-key infrastructure (PKI) is used in Internet voting schemes for establishing secure communication channels between the voters and the voting system components. In particular, it enables authentication via digital signatures used by the voters and other entities involved in the election to digitally sign their messages. Additionally, in case a scheme requires private channels between the voters, or between voters and other entities involved in the election, the PKI is used to facilitate end-to-end encryption of the messages that are being exchanged.

2.2.15 Decentralized Key Exchange With Short Authentication Strings

For establishing the public-key infrastructure without relying on centralized certificate authorities, a method for decentralized key exchange has been developed [NR06] that can be used by groups of participants to exchange their public signature keys. The security of the scheme relies on short authentication strings and an out-of-band channel. Namely, at the end of the exchange, each participant computes a short hash value h_i of the other participants public signature keys and other values she has received. If no man-in-the-middle attack occurred, each participant should get the same h_i. These values are then manually compared over an out-of-band channel, which might be a video call or communication via physical proximity. Note, that the necessity of manual comparison over such a channel

implies, that the decentralized key exchange according to this scheme is only feasible for relatively small groups of participants.

A variant of the scheme was implemented by Farb et al. in their smartphone application SafeSlinger [FBC+12]. For the sake of better usability, SafeSlinger presents the 24-bit hash values that the participants have to manually compare as passphrases of three words, constructed according to the PGP Word List [Zim95].

2.2.16 Diffie-Hellman Key Exchange

For jointly generating a common secret key between two participants, the key exchange scheme has been proposed by Diffie and Hellman in [DH76]. The key exchange proceeds as follows: given a common value $g \in \mathbb{G}_q$, each one of the participants generates a secret value $x_i \leftarrow_\$ \mathbb{Z}_q$, and sends $Y_i = g^{x_i}$ to another participant. When receiving a value $Y_j = g^{x_j}$ from another participant j, the participant i calculates $K = Y_j^{x_i} = g^{x_i x_j}$. The value K is then used as a symmetric secret key for encrypting the messages, communicated between i and j.

Note, that in order to prevent man-in-the-middle attacks, it is important that the messages Y_i, Y_j are send in an authenticated manner. Hence, the public signature keys of both the participants have to be exchanged beforehand.

2.2.17 Plaintext Equivalence Tests

Plaintext equivalence tests (PET) [JJ00] are used in order to check, whether two ciphertexts encrypt the same plaintext without revealing any more information about the plaintexts or their relation to each other.

For a pair of ElGamal ciphertexts $c, c' \in \mathbb{G}_q^2$ with pk as a corresponding public key, $c = \mathsf{Enc}(\mathsf{pk}, m) = (a, b), c' = \mathsf{Enc}(\mathsf{pk}, m') = (a', b')$, these tests are performed in a distributed way by a group of trustees that own the shared corresponding private key. The trustees compute and jointly decrypt

$$\left((\frac{a}{a'})^z, (\frac{b}{b'})^z \right)$$

for a jointly generated random secret z.

The result is the value of $(\frac{m}{m'})^z$ which is 1 if $m = m'$, or a random value in $\mathbb{G}_q$ that reveals no information about m, m' or their relation to each other otherwise.

Alternatively, for testing whether a ciphertext c encrypts a message m without revealing any additional information about the plaintext of c, the PETs are performed on the ciphertexts c, c' with c' as an encryption of m with a public randomness value (for example, for the ElGamal public key (g, h), $c' := (g, m \cdot h)$).

2.2.18 Byzantine Agreement

A number of so-called Byzantine agreement schemes proposed in the literature are designed to solve the problem with consistent communication in a decentralized setting, where some of the communication parties are assumed to be faulty. In particular, the proposal in [LSP82] requires authenticated messages (for example, via digital signatures) and ensures consistent communication in case more than half of the parties are honest. Note, however, that this proposal has a high level of round complexity: given f faulty parties, broadcasting one message via Byzantine agreement requires $f + 1$ rounds of communication.

2.3 Helios

In this section we provide an overview of Helios and its existing extensions and describe the version of Helios that our extensions are based upon.

Helios is a well-established voting system, originally developed by Adida and described in [Adi08]. The open-source implementation of Helios has been used in several real-world elections, e.g. the elections of the International Association for Cryptologic Research [IAC16] or the University president election at UC Louvain [ADMP+09]. The scheme behind Helios has furthermore been extensively studied in literature [KTV12, BCG+15, BPW12, KZZ16], whereby formal proofs for its security has been provided [BPW12, BCG+15, CGGI14, KRS10].

2.3.1 Overview

The basic idea of Helios, as described in [Adi08] utilizes the cryptographic techniques mentioned in Section 2.2. The scheme can be described as follows: The *registration authority* generates and distributes the login credentials (as usernames and passwords) to eligible voters. The *tabulation teller* generates an ElGamal key (Section 2.2.1) that is used in the election (further referred to as an *election key*) and publishes the public part of it. For voting, the voters use their primary *voting device*s to construct their ballots by encrypting a chosen voting option with the public election key published by the tabulation teller. They then have an to either cast the ballot by authenticating themselves with their username and password to the *bulletin board* and submitting their ballot to it, or to verify the ballot with an *verification device* using the Benaloh challenge [Ben06] in order to ensure, that the ballot encrypts the intended voting option. The Benaloh challenge works as follows: If the voter decides to verify, the voting device outputs the randomness used in encrypting the vote. The user can then use the verification device to verify, that encrypting her chosen voting option using the output randomness results in the same ciphertext that was computed as their ballot. Once verified, the ballot can no longer be cast, so the voter has to construct a new ballot afterwards. The voter can choose to verify as many ballots as she wants, and the more she does it, the better assurance the verification pro-

vides. After the voter casts her ballot instead of verifying it, it is published on the bulletin board near the voter's username. The voter can then verify whether her ballot has been correctly stored by the voting system. After the voting is finished for all the voters, the cast ballots are anonymized by the tabulation teller using a verifiable re-encryption mix net (Section 2.2.12)[3]. After the anonymization, the resulted ciphertexts are decrypted by the tabulation teller who also provides the proof of decryption validity together with the decryption result (Section 2.2.7).

2.3.2 Helios Extensions

Several extensions has been proposed for Helios, improving its security and usability as well as introducing additional functionality.

As such, several extensions of Helios focus on improving *vote integrity* by introducing new ways that allow the voter to verify that their vote has been encrypted and stored correctly. The Zeus voting system [TPLT13], used in University of Athens election, modifies Helios by introducing an additional way to verify that the voting device encrypts the voting option intended by the voter. Namely, the voters have an option to cast a ballot using verify codes distributed to them at the registration, so that the ballots cast with those codes are not included in the tallying, but decrypted instead, so that the voters could verify their correctness. Further methods for the voters to verify the vote integrity of their vote have been proposed in the Selene protocol [RRI15], which introduced tracking number appended to the cast ballots, and Guasch et al. [GM16, EGHM16] employing designated-verifier proofs. The proposal by Bernhard et al. [BPW12] improves the vote integrity of Helios by improving the soundness of the proofs of knowledge used in Helios via using strong Fiat-Shamir heuristic. Their proposal also introduces the proof of plaintext knowledge that is included with the ballot for improving vote privacy by preventing ballot copying attacks described in [SB13]. The Apollo extension of Helios introduces the usage of voting assistants to improve the voter-side verifications in [GKV+16].

The extension by Cortier et al. improves *eligibility* by requiring the voters to digitally sign their ballots upon voting [CGGI14]. Another extension by Srinivasan et al. [SCH+14] also focuses on improving eligibility by using a novel cryptographic primitive, token-based encryption.

Further extensions to Helios has been focused on improving the vote privacy requirement. As such, the BeleniosRF protocol [CFG15] introduces receipt freeness into Helios by using a novel cryptographic primitive, signatures on randomizable ciphertexts. The extension ensuring everlasting privacy was proposed by Demirel et al. [DVDGdSA12]. Note, however, that while the authors of [DVDGdSA12] present their proposal as an extension of Helios, they reuse very few components from the original system by using a different way to encrypt the votes (namely, Pedersen commitments published on the bulletin board

[3] Note, that subsequent versions of Helios use homomorphic tallying as opposed to mix net for anonymizing the ballots.

and Pallier encryption for encrypting the decommitment values sent to the tabulation tellers over private channels) for voting and a modified process for tallying. A subsequent version of Helios, called Helios 2.0 [ADMP+09], improves vote privacy and fairness by sharing the private election key among multiple tabulation tellers, so that the private election key key could be calculated as a sum of each tabulation teller's share. A further extension [CGGI13] relies on the proposal in [CGS97] adjusted to Helios and improves vote privacy as well as robustness by introducing distributed tallying via Pedersen secret sharing.

An extension with improvement of robustness has also been proposed in [CBP16]. The proposal focused on preventing the denial-of-service attacks by distributing the bulletin board amoung multiple parallel servers.

From the usability perspective, a number of suggestions that simplify the verification process in the current Helios implementation for the voters, have been proposed [NORV14, KKO+11].

Further proposals suggested adding functionality to Helios. As such, the proposal of Desmedt et al. [DC12] introduced blind ballot copying using divertible proofs, which enabled the voters to request a copy of the ballot cast by another voter and cast it as their own in blinded form. The proposal in [PR16] further extended Helios towards the support of the new form of voting, the so-called quadratic voting.

2.3.3 Helios-Base

In the following, the variant of Helios that is used as a base for our extensions (further referred to as Helios-Base) is described.

Pre-Considerations

We make the following modifications to the scheme described in Section 2.3.1:

1. For improving eligibility, we make the voters digitally sign their ballots before casting them, similar to the suggestion in [CGGI14].

2. For better flexibility we allow both the option of using mix net (Section 2.2.12) with each tabulation teller acting as a mix node, or homomorphic tallying (Section 2.2.9) approach for anonymizing the ballots. Depending on the chosen approach, different proofs of knowledge are used throughout the election: the proofs of 1-out-of-L encryption (Section 2.2.6) are crucial for the homomorphic tallying approach, and the proof of shuffle validity is required for the mix net approach.

3. For improving vote privacy, we make the voters submit the proof of plaintext knowledge (see Section 2.2.5) with their ballot, as proposed in [BPW12]. Furthermore, as also proposed in [BPW12], for improving vote integrity we construct all the proofs

of knowledge that are used throughout the election (i.e. proof of 1-out-of-L encryption or proof of shuffle validity, depending on the anonymization method, proof of plaintext knowledge and proofs of decryption validity (Section 2.2.7)) using strong Fiat-Shamir heuristic (Section 2.2.2).

4. For improving vote privacy and robustness, we distribute the election key generation between multiple tabulation tellers via distributed threshold secret sharing and make the tabulation teller decrypt the ballots via distributed threshold decryption (Section 2.2.11), as proposed in [CGGI13].

5. For improving vote privacy, we further require that the tabulation tellers verify that the bulletin board has published the correct public election key as proposed in [KZZ16].

In this way, a scheme that incorporates these extensions is more secure than the original Helios, yet retains its flexibility, making it suitable as a basis for further extensions. We elaborate on our choices and provide more details about them below.

Use of Digital Signatures. Following the proposal by Cortier et al. [CGGI14], digital signatures are used for authenticating the voters. This proposal distributes the trust in ensuring, that only the digital signatures from eligible voters are accepted, between the registration authority and the bulletin board. Namely, it requires the registration authority to generate the signature keys for the voters, and in order to prevent distribution to these keys to non-eligible voters, the bulletin board distributes a set its own login credentials to the voters. Thus, the voters both have to digitally sign their ballots with the keys received from the registration authority, and authenticate themselves to the bulletin board with their login credentials.

In order to provide more control to the voters, however, instead of letting the registration authority and bulletin board distribute the signature keys and login credentials to the voters, in our version of Helios-Base we rely on a trusted public key infrastructure (PKI, Section 2.2.14). This PKI is assumed to be tied to the voter register, so that a public signature key of each voter is available through the PKI, and only the voter knows her corresponding private signature key. The PKI can be established in one of the following ways. The first way is to use an existing PKI (such as national eID as in Estonia or Germany), which is independent from the election. The second way is to make the voters generate their signature key themselves and submit the public signature keys to the registration authority who in turn publishes it on the bulletin board. Unless mentioned otherwise, both of these variants can be employed in our extensions. For the sake of enabling the verification that only eligible voters participate in the election, the public signature keys of the voters are publicly linked to the voters identities.

Note, that due to the usage of a PKI that is coupled to the voter's real identities, Helios-Base does not ensure the participation privacy requirement, as opposed to [CGGI14] that

advices using pseudonyms for the voters in case better privacy is required. However, this is a trade-off that we make for the sake of stronger eligibility guarantees. As such, if pseudonyms are used instead of the real voters identities, then the entity that assigns these pseudonyms should be trusted to only assign them to eligible voters. Instead, we presume that using real identities would make it easier to verify, that these identities and the corresponding public signature keys belong to eligible voters. Indeed, as mentioned by Pereira in [Per16], one of the ways to check for possible manipulations is to contact the voters to ensure that they are eligible to vote in this election or to check whether they verified that their ballot is stored correctly.

Anonymizing the Ballots. For anonymizing the ballots, either the homomorphic tallying approach (also used in [CGGI14] and in subsequent versions of Helios starting from Helios 2.0 [ADMP+09]) or the mix net approach (used in the original proposal in [Adi08]) with each tabulation teller acting as a mix node in a verifiable re-encryption mix net is used. Both of these approaches have their advantages and disadvantages. The homomorphic tallying approach is much more efficient in case of elections with small size of electorate and simple voting rules, such as a yes/no referendum. The mix net approach, on the other side, allows conducting elections with more complex rules, including write-in ballots that would be impossible to tally using the homomorphic tallying approach. Note, that the particular choice of the anonymization method does not affect the security of the scheme, assuming the reliability of the corresponding cryptographic primitives and a computationally restricted adversary.

Use of Strong Fiat-Shamir Heuristic and Proofs of Plaintext Knowledge. The authors of [BPW12] suggested adding a proof of plaintext knowledge to the ballot, for countering the ballot copying attack which could lead to a violation of vote privacy. They furthermore have shown, that an election can be easily manipulated by either malicious voters or malicious tabulation tellers with falsifying the proofs of knowledge, such as proofs of 1-out-of-L encryption (Section 2.2.6) and proofs of decryption validity (Section 2.2.7) as originally used in Helios. In order to mitigate this attack, they suggested using strong Fiat-Shamir heuristic for constructing the non-interactive proofs of knowledge. Hence, we use their proposed extension in Helios-Base by requiring the voters to submit the proof of plaintext knowledge with their vote, and by constructing all the proofs of knowledge used in Helios-Base (proof of plaintext knowledge, proof of 1-out-of-L encryption in case of homomorphic tallying, proof of shuffle validity in case of mix net and proof of decryption validity) using their suggestion. Note, that both of these extensions have also been incorporated in [CGGI14].

Multiple Tabulation Tellers. While the original Helios only used one tabulation teller, the subsequent versions starting from Helios 2.0 distributed the trust regarding vote privacy by enabling joint generation of the election key among multiple tabulation teller

[ADMP+09], so that each tabulation teller possesses a secret share of the private election key. This approach, however, while improving vote privacy, suffers from a drawback in robustness, since a single missing share of the private election key would make it impossible to tally the election. Hence, our version of Helios-Base uses the proposal in [CGGI13] which improves both vote privacy and robustness of the original Helios by using distributed threshold secret sharing and distributed threshold decryption for distributively generating the election key between the tabulation teller and distributed threshold decryption for tallying the votes.

Verification by the Tabulation Teller. The authors of [KZZ16] mention, that the tabulation tellers in the original Helios are not instructed to verify the correctness of the public election key as published on the bulletin board. This, however, could lead to a man-in-the-middle attack whereby an adversary controlling the bulletin board publishes her own public election key instead, thus being able to decrypt all the ciphertexts submitted with the ballots and violate vote privacy for all the voters. Hence, we require that the tabulation tellers verify that the bulletin board publishes the data as submitted to it during the election key generation.

Description

We further describe the election process in more details as follows. For the sake of simplicity, we describe the single choice ("yes/no") election, where the voters cast either 1 or 0 (represented as g^0 or g^1) as their vote, although a generalization to more complex ballots is possible. The following entities are involved in the protocol:

- *Election organizers*, responsible for publishing the general information for the election, incl. the voting options. [4]
- *Registration authority*, responsible for maintaining the so-called *voting register*, which is a publicly available list of eligible voters' public signature keys,
- *Bulletin board*, acting as a public append-only broadcast channel that is used for publishing all necessary election information and cast ballots,
- *Tabulation tellers*, responsible for generating the election key, anonymizing the cast ballots and decrypting the result.

The voter environment consists of a voting device, used for casting the ballot, and a verification device used for verifying that the ballot encrypts a correct voting option. The components of Helios and the interactions between them is depicted on Figure 2.1.

The election process can be outlined as follows.

[4] Note that since the published information can be easily verified by the parties involved in the election, we do not consider the election organizers in our security models.

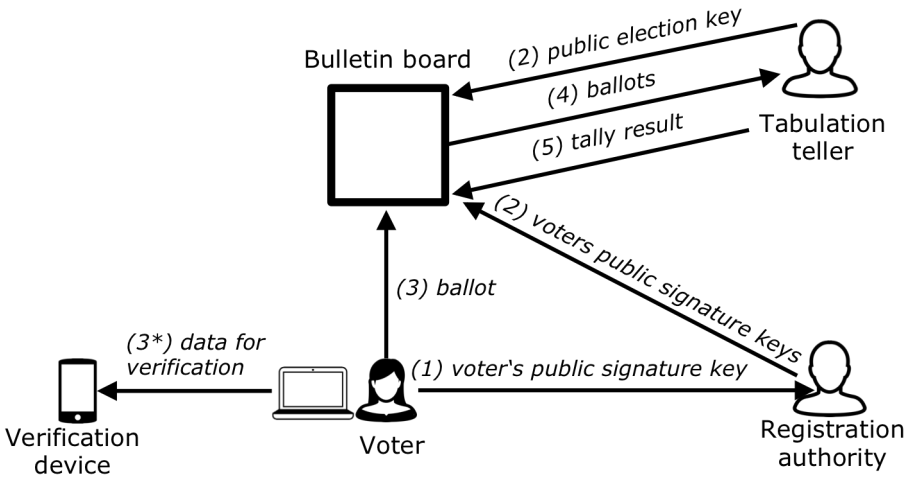


Figure 2.1: Components and their interactions of Helios, numbered by the order of execution steps. Note, that verifying a ballot (3*)) is optional, and sending the voter's public signature key can be omitted if an existing PKI is used.

Setup. If there is no existing PKI with the public signature keys of the voters that can be used for the election, the voters generate and submit their public signature keys to the registration authority. The registration authority publishes these public signature key on the bulletin board near the voters identities, and the voters verify that their public signature keys have been published correctly. In case of a pre-existing trustworthy PKI, the registration authority uses it to publish the identities of the eligible voters and their public signature keys on the bulletin board. In that case, one can always use the existing PKI to verify, that the published public signature keys are correct. The N_t tabulation tellers jointly generate an election key via distributed threshold secret sharing (Section 2.2.11) with $\mathsf{pk} = (g, h = g^s) \in \mathbb{G}_q^2$ as the public election key, and the private election key sk distributed into N_t shares with $t = \lfloor N_t/2 \rfloor$ as the threshold. The public election key $\mathsf{pk} = (g, h)$ as well as the other data produced during the election key generation that is required for the proofs of decryption validity, is published on the bulletin board. The setup is concluded by publishing the list of valid voting options $\{v_1 = g^0, v_1 = g^1\} \subset \mathbb{G}_q$.

Voting. In order to vote for a voting option $v \in \{v_0 = g^0, v_1 = g^1\}$, the voter id_i prepares her ballot $(c = \mathsf{Enc}(\mathsf{pk}, v), \pi_v)$ with:

- $c = (a, b) = (g^r, vh^r)$ as the encryption of a voting option v using the public election key pk,
- $\pi_v = PoK\{r \in \mathbb{Z}_q : a = g^r \wedge (b = v_0 h^r \vee b = v_1 h^r)\}$ as the proof of well-formedness, used to prove the plaintext knowledge of v (Section 2.2.5) and that $v \in \{v_0, v_1\}$ is a valid voting option (Section 2.2.6). In case the mix net approach is used for

anonymizing the ballots, the proof of well-formedness can be simplified to only proving the plaintext knowledge of v.

After preparing a ballot, the voter has an option either to cast it by submitting it to the bulletin board, or to verify the ballot using the Benaloh challenge [Ben06] as in the original version of Helios (see Section 2.3.1). The purpose of the verification is to ensure, that the ballot was prepared correctly by the voting device. The voter can verify as many ballots as she wants, however, once verified, the ballot can no longer be cast. When the voter decides to cast her ballot, she digitally signs it with her private signature key and submits it to the bulletin board. After casting the ballot, the voter verifies the bulletin board by checking whether the ballot is correctly posted there.

Note, that for preventing the man-in-the-middle attacks, it is important for the voter to verify that she communicates with an authentic bulletin board while casting her ballot or verifying that it is properly published.

Tallying. After the voting has finished, the bulletin board removes all duplicate ballots and ballots with invalid proofs of knowledge. In case the election allows vote updating, out of all the ballots cast by the same voter, only the last ballot is kept. The voters can once again verify, that all their ballots are properly stored on the bulletin board before the tallying begins. Prior to the decryption, the ballots have to be anonymized. If the mix net approach is used for the anonymization, the ciphertexts from the cast ballots are shuffled using a verifiable re-encryption mix net with each tabulation teller acting as a mix node. If the homomorphic tallying approach is used, the ciphertexts are multiplied together to form an encryption of the sum of all the votes.

After the ballots have been anonymized, the result of the anonymization is decrypted by the tabulation tellers via distributed threshold decryption and published. The proofs of shuffle validity (in case of the mix net anonymization) and the proofs of decryption validity, as well as digital signatures and proofs of well-formedness submitted with the ballots during voting, are published to enable verifying that the votes have been tallied correctly.

Security Model

We describe the security model for Helios-Base by listing the security assumptions under which the security requirements from Section 2.1 are satisfied (based upon the results in [BPW12, BCG+15, CGGI14, CGGI13, KZZ16, LSBV10] and our informal evaluation). Here and in the further descriptions of the security models and the security evaluations in the thesis, we consider an entity in the voting system (excluding the voters) to be honest if she follows the scheme and neither divulges her private input to the adversary or to the public, nor uses it herself in an unauthorized way, e.g. by attempting to decrypt a ciphertext she is not authorized to. The voters, on the other hand, are considered honest

if they are not under complete adversarial control, however, might still deviate from the prescribed behavior during the election, e.g. in case of coercion.

Vote Privacy. Vote privacy is preserved under the assumptions, that more than half of the tabulation tellers are honest, the voting devices do not leak the voters chosen options to an adversary, the adversary is computationally restricted, the bulletin board does not remove or modify the data published on it and shows the same contents to everyone, the voter verifies that she communicates with an authentic bulletin board while casting the ballot, and the adversary does not coerce the voters into casting a vote for a specific voting option. Note, that the last assumption means, that neither receipt-freeness nor coercion-resistance is ensured in Helios, and the assumption of computationally restricted adversary means that everlasting privacy is not ensured as well.

Fairness. Fairness is preserved under the same assumptions as vote privacy.

Participation Privacy. Participation privacy is not ensured in Helios-Base, since the identities of the voters who cast their ballots are public.

Eligibility. Eligibility is preserved under the assumptions, that the voting register is trustworthy, the adversary is computationally restricted and that the devices of honest voters do not leak the voters private signature keys to the adversary.

Vote integrity. Vote integrity is preserved under the assumptions, that either the voting device or the verification device of honest voters is trustworthy, the bulletin board shows the same contents to everyone, the adversary is computationally restricted, and the voters perform the necessary verifications.

Robustness. Robustness is preserved under the assumption, that the majority of the tabulation tellers are honest and produce the required output during the tally, and that the contents of the bulletin board are available for tally (i.e. the bulletin board does not delete the published data and shows the same contents to everyone).

The resulting list of assumptions is thus as follows:

(A-H-TabTellerHonest) More than half of tabulation tellers are honest and capable of communicating with each other and the bulletin board.

(A-H-VotDeviceLeakage) The voting devices[5] of voters do not leak data to an adversary.

[5] Here and in the rest of security evaluations in the thesis we refer to voting device as a set of all hardware and software components, including the voting application, that is used by the voters for casting their ballots.

(A-H-NoBBModification) The bulletin board does not remove or modify the data published on it.

(A-H-BBConsistency) The bulletin board shows the same contents to everyone.

(A-H-NoCoercion) No coercion or vote selling takes place.

(A-H-CompRestricted) The adversary is computationally restricted.

(A-H-Verify) The voters perform the verifications available to them within the system.

(A-H-VerDeviceTrusted) The verification devices[6] of the voters are trustworthy.

(A-H-VotRegister) The voting register, with the eligible voters public signature keys either generated for a specific election or based on a pre-existing PKI, as published on the bulletin board is trustworthy. Note, that in case the voters public signature keys have been generated for a specific election and are only available on the bulletin board, the assumptions that the bulletin board does not delete or modify its contents (A-H-NoBBModification), shows the same view to everyone (A-H-BBConsistency), and the voters verify the correctness of their published public signature keys are also required.

We hence aim to preserve the security model in our extensions, deviating from it only if justified by the setting.

[6] As with voting devices, we refer to the verification device as a set of all hardware and software components used for the voters for verifications.

Chapter 3

New Voting Setting: Boardroom Voting

Much of the current research on Internet voting has been focused on large scale elections, such as political elections. Still, there are also many small-scale elections, such as voting in private associations, committees and boards of directors. While currently these elections are mostly conducted via paper ballots or simple show of hands, Internet voting would also allows some of the voters participating remotely. We refer to elections based in such setting as *boardroom voting*.

Our contribution in this chapter is to extend Helios-Base (Section 2.3.3) towards boardroom voting. We also make suggestions on how to ensure that the faults that might occur during the election in boardroom voting setting, such as non-responding or malicious participants, or inconsistent communication, are properly addressed in our extension.

This chapter is structured as follows. In Section 3.1 the requirements are listed, specific to boardroom voting. Further, the security model relevant for boardroom voting is described in Section 3.2. The proposed scheme is described in Section 3.3, and its security is evaluated in Section 3.4. The related work on boardroom voting schemes is described in Section 3.5. The summary of the chapter and the future work is outlined in Section 3.6.

Parts of this chapter have been published at the *6th International Conference on Electronic Voting, Verifying the Vote* [KNV+14].

3.1 Setting Requirements

In this section, we consider the differences between boardroom voting and large-scale elections and derive boardroom-specific requirements from these differences.

Large-scale elections tend to appoint trusted entities for security-critical tasks in the election. These entities are chosen so that they represent different interest groups, for example, competing political parties. In this way, their malicious collaboration is assumed to be unlikely. On the contrary, the interest groups are not always so explicitly defined in boardroom voting. Hence, choosing and appointing such entities is not always feasible in such a setting. However, the much smaller size of the electorate enables an efficient implementation of full trust distribution, i.e. between all the voters who take over all

the security-critical tasks. Correspondingly, the first boardroom-specific requirement is as follows:

Decentralization. For any security requirement, the trust should be distributed among the voters in the boardroom voting election.

Large-scale elections tend to be prepared well in advance, including the list of eligible voters and their public signature keys. Boardroom voting is often performed in an ad-hoc fashion: the decision to vote on some issue might spontaneously arise during the meeting. Furthermore, the group of board members may change on a regular basis, for example, the board members might be represented by different people in different meetings. Thus it is not known in advance, whether there will be an issue that has to be voted on, and which board members will participate. Correspondingly, the second boardroom voting requirement is as follows:

Ad-hoc Elections. It should be possible to decide during the meeting, whether there is voting that should be conducted during the same meeting, and which voters should be eligible to participate.

3.2 Security Model

In this section, we describe the assumptions, under which the security requirements should be ensured in our extension. In our extension, we aim to preserve the security model of Helios-Base, with the exception of the constrains dictated by the boardroom voting setting. Namely, the security requirements from Section 2.1 must hold under the following assumptions:

Vote Privacy. Vote privacy should be ensured under the assumptions that more than half of all the voters are honest, the voting devices of the honest voters are trustworthy, the adversary is computationally restricted, and the adversary is not capable of coercing the voters.

Fairness. Fairness should be ensured under the same assumptions as vote privacy.

Participation Privacy. Following Helios-Base, and since the identities of boardroom members that participate in the meeting are public, our extension does not ensure participation privacy.

Eligibility. Eligibility should be ensured under the assumption that an adversary is computationally restricted, the identities of eligible voters are known to all the other voters, and the voter's private signature key is not leaked to the adversary by the voting device.

Vote Integrity. Vote integrity should be ensured under the assumptions that the devices of the honest voters are trustworthy and that the adversary is computationally restricted.

Robustness. Robustness should be ensured under the assumptions that more than half of all the voters are honest, their devices are trustworthy, and they are able to communicate with each other during decryption, and that the adversary is computationally restricted.

The aforementioned assumptions can be summarized as follows:

(A-BV-HalfVotersHonest) More than the half of all the voters are honest and available during the whole voting process, i.e. during voting and tallying.

(A-BV-Communication) The devices of honest voters are able to communicate with each other.

(A-BV-NoCoercion) No coercion or vote selling takes place.

(A-BV-CompRestricted) The adversary is computationally restricted.

(A-BV-VotDeviceTrusted) The devices of honest voters are trustworthy.

(A-BV-EligVotersKnown) All the voters know, which other voters are eligible to participate in the election.

The assumptions (A-BV-HalfVotersHonest), (A-BV-Communication), (A-BV-NoCoercion) and (A-BV-CompRestricted) are are the same as the assumptions regarding the tabulation tellers for Helios-Base. Further assumptions that are required for our extension but not for Helios-Base itself are explained as follows:

(A-BV-VotDeviceTrusted) While it would be theoretically possible for the voters to use a second device for verifying their votes, this would pose more difficulties in the boardroom voting setting. As such, the voters have to make sure that they have their second devices (i.e. a second smartphone) next to them during the voting, which is not always a given due to ad-hoc nature of elections. Furthermore, as the verification procedure has to be conducted several times for better security, this might pose difficulties to the voters due to the limited time appointed for voting.

(A-BV-EligVotersKnown) This assumption is required for conducting the voter registration in a decentralized way, and is justified in a boardroom voting setting due to the small amount of voters in the election.

3.3 Description

In this section we describe the scheme extending Helios-Base towards boardroom voting setting. From the requirements outlined in Section 3.1, the following challenges can be derived: The first challenge is due to the fact that a central registration authority cannot be assumed in boardroom voting, due to the decentralization requirement. Furthermore, the requirement of ad-hoc elections requires an approach that allows the voters to reliably exchange their public signature keys during the meeting, i.e. without significant preparations beforehand and in a timely manner. Hence, a way for the votes to reliably exchange their public signature keys in a decentralized ad-hoc fashion is needed. The second challenge is to distribute the task of the bulletin board due to the decentralization requirement. For this purpose, a broadcast channel to enable the communication between the voters should be established, which should be reliable even in presence of some malicious voters.

Recall, the Helios-Base scheme works as follows. In the setup phase, the registration authority publishes the public signature keys of the eligible voters, either sent by the voters for the particular election, or taken from a pre-existing PKI, on the bulletin board. The tabulation tellers furthermore jointly generate an election key via distributed threshold secret sharing (Section 2.2.11) and publish it on the bulletin board together with the data required for verifying the proofs of decryption validity. In the voting phase, the voters cast their ballot by encrypting their preferred voting option and computing the proof of well-formedness (Section 2.2.6 or Section 2.2.5, depending on which anonymization method is used). After computing the ballot, the voters have an option to verify that it encrypts their intended voting option (the ballot is then discarded) or digitally sign the ballot and send it to the bulletin board. After the voting is finished, the tabulation tellers take the cast ballots that are published on the bulletin board, discard the ballots with invalid proofs of well-formedness, and anonymize the rest of the ballot using either mix net (Section 2.2.12) or homomorphic tallying approach (Section 2.2.9). Afterwards, the tabulation tellers jointly decrypt the anonymized result via distributed threshold decryption.

In our extension to the Helios-Base, voters take over the tasks performed by the various entities in Helios-Base. More precisely:

- In their role as registration authority, voters generate their public signature keys and run a decentralized key exchange scheme (Section 2.2.15) for reliably exchanging these keys. The reliability of the decentralized key exchange is ensured via the manual verification the passphrases that serve as short authentication strings. Afterwards, the voters use the exchanged public signature keys in Diffie-Hellman key exchange Section 2.2.16 to generate symmetric secret keys which are later used for encrypting messages and establishing private communication channels between the voters.

- In their role as tabulation tellers, voters jointly generate an election key via distributed threshold secret sharing (Section 2.2.11). Once the voting is finished, vot-

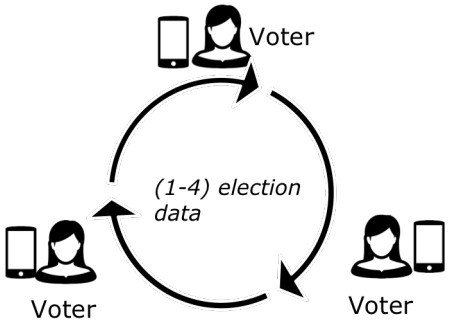


Figure 3.1: Components and their interactions of Helios-BV. The voters run the election between themselves by exchanging the election data in (1) setup via decentralized key exchange, (2) election key generation, (3) voting, and (4) tallying.

ers either jointly mix the cast ballots with a verifiable re-encryption mix net (Section 2.2.12), or multiply the ballots to get a ciphertext of the sum of cast votes, following the homomorphic tallying approach (Section 2.2.9). The result of the anonymization (i.e. either individual shuffled ballots, or the sum) is decrypted with distributed threshold decryption.

- In their role as bulletin board, voters broadcast all their messages via decentralized communication between them. The reliability of the decentralized communication is ensured via Byzantine agreement as described in Section 2.2.18.

Note that one of the voters acts as an election organizer. This role does not possess additional privileges. The task of an election organizer lies in initiating the election by supplying the necessary information such as a question that is voted on, and starting the scheme execution. For the distributed threshold decryption we set the threshold as $t = \lfloor N/2 \rfloor + 1$ for N as a total number of voters, since otherwise a malicious minority of voters can compromise either vote privacy (given $t < \lfloor N/2 \rfloor + 1$) or robustness (given $t > \lfloor N/2 \rfloor + 1$). The components and their interactions of our extension are depicted at Figure 3.1.

We further take following approach to address the faults that might occur during the election. We have identified the steps in the executing of the voting process, whereby some faults might be present, most commonly some voters not being present or able to communicate with the others. Some of the cryptographic primitives used in the scheme are already designed to handle some of these faults. As such, in case some voter fails to produce a valid shuffle result (if the mix net approach is used for the anonymization), the output of a previous voter is being processed further. Furthermore, as will be shown in Section 3.4, some of these faults, such as the voters failing to produce valid partial decryptions of the ballots, could be ignored under the assumptions that we made.

Other faults are the ones that occur during phases that preclude the tallying. Namely,

faults could be present during the decentralized key exchange (i.e. the adversary trying to execute a man-in-the-middle attack), ballot initialization stage (such as voters not responding to the invitation to vote), or voting. The diagrams in Figures 3.2a to 3.2c show the way the scheme is supposed to handle these faults. As such, for example, the voter who wishes to initiate the election has the option to decide, whether she still wants to start the election if not all of the invited voters respond to her invitation, or to wait some more for the missing voters to respond, or to cancel the election.

Another source of faults during the voting, is the inconsistency of message broadcast. If, instead of being broadcast, the message is sent separately to each receiver, it makes the communication vulnerable to Byzantine faults. Namely, a malicious voter can send different messages to different receivers (for example, during broadcasting a cast ballot), thus endangering robustness. These faults, however, are properly handled in case when the Byzantine agreement is used in the communication between the voters.

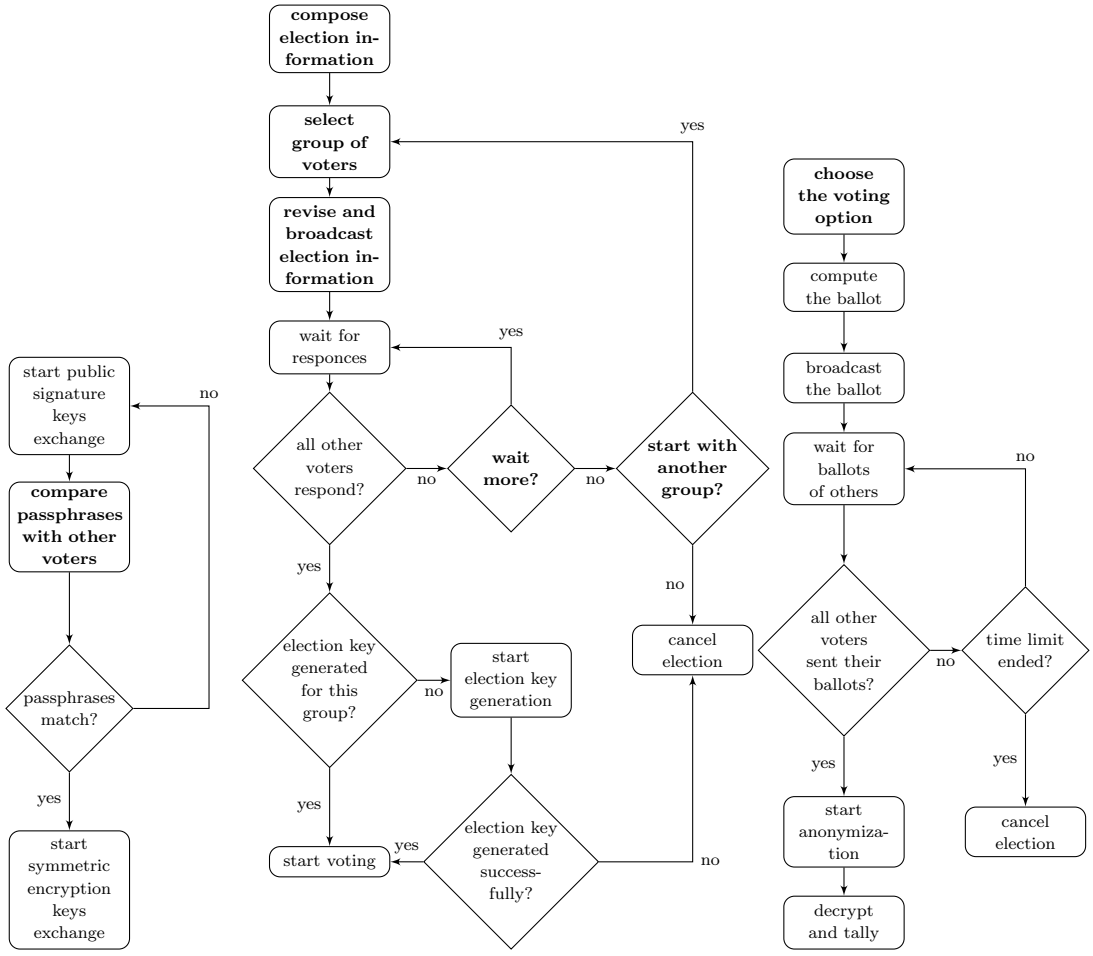
3.4 Security Evaluation

This section is dedicated to an informal security argument on the presented scheme. To evaluate its security according to the security model described in Section 3.2, we identify the threats against the security requirements [7] and show that the scheme defends against these threats under the given assumptions. For this purpose, we study each step of the scheme in order to consider the possibilities for the adversary to intervene and either get the information that is meant to be private or modify the data communicated or computed within the election. Note that similar approach has been used in other works, such as [LSBV10].

Vote Privacy. Breaking vote privacy would imply establishing a link between the plaintext vote that is revealed at the end of tallying and the identity of the voter who submitted the corresponding ballot. We consider the different steps of the election at which it could be done.

The proposed fault handling ensures that the voters have an option to decline participating in the election, if they do not trust the majority of the voters to be honest. Since the adversary is able to find ways to break vote privacy if she manages to impersonate the honest voters and conduct a man-in-the-middle attack, we start off by arguing that such impersonation of the voters is infeasible under the assumptions given in Section 3.2, namely, (A-BV-NoCoercion), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted). Due to the proposed fault handling, the election does not start unless all the voters confirm that the decentralized key exchange has been performed correctly. The decentralized key exchange scheme thus ensures that as long as the same passphrase is output for all the voters and the

[7] We acknowledge that in absence of formal proofs the list of such threats is not guaranteed to be exhaustive.



(a) Decentralized Key exchange **(b)** Election Initiation **(c)** Voting

Figure 3.2: Fault handling in different stages. The bold text denotes the steps where voter's input is required, e.g. as a decision that a voter needs to make.

adversary is incapable of finding a collision for the hash function used in the decentralized key exchange (A-BV-CompRestricted), all the voters have the valid public signature keys of other voters. Then, unless the private signature key of the voter is leaked by themselves in case of coercion (prevented by the assumption (A-BV-NoCoercion)) or by their malicious device (A-BV-VotDeviceTrusted), or the adversary manages to forge a digital signature (prevented by the assumption (A-BV-CompRestricted)), voter impersonation is infeasible.

We now consider other ways for the adversary to break vote privacy. The voting and the tallying proceed in the same way as in Helios-Base, hence a similar argument for their security holds. During voting, the vote is encrypted at the time that it is submitted by the voter with attached voter's identity. Revealing its plaintext value at this

stage would require decrypting the ciphertext, which is possible either via breaking the security of the encryption (prevented by the assumption (A-BV-CompRestricted)), getting the randomness value used in encrypting the ballot either from the voter herself in case of coercion (prevented by the assumption (A-BV-NoCoercion)) or from the corrupted voter's device (prevented by the assumption (A-BV-VotDeviceTrusted)), or in getting the private election key. The election key generation scheme ensures that the private election key cannot be reconstructed, unless the adversary either corrupts more than half of the voters or their devices (prevented by the assumptions (A-HalfVotersHonest) and (A-BV-VotDeviceTrusted)), gets access to the private communication channels between the honest voters either by breaking the symmetric encryption scheme used to encrypt the messages sent over those channels (prevented by the assumption (A-BV-CompRestricted)) or breaking the security of Diffie-Hellman key exchange used to generate symmetric secret keys (A-BV-CompRestricted) or impersonates the voters (as shown above, prevented by the assumptions (A-BV-NoCoercion), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted)). Furthermore, the ballot copying attacks are also prevented due to the well-formedness proofs (A-BV-CompRestricted).

At the tallying stage, the anonymization procedure via homomorphic tallying ensures that only the sum of all the votes is being decrypted. Alternatively, if the mix net approach is used, it ensures that the link between the non-anonymized ciphertexts and their plaintext value cannot be established, unless all but one[8] voter reveal their correspondences between input and shuffled ciphertexts. Hence, the adversary can prevent the ballots from being anonymized via mix net only in case she corrupts more than $N-2$ voters or their devices (which is prevented by the assumptions (A-BV-HalfVotersHonest) and (A-BV-VotDeviceTrusted)), or impersonates the honest voters (prevented by the assumptions (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted)).

Thus, vote privacy is ensured under the assumptions (A-BV-HalfVotersHonest), (A-BV-NoCoercion), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted).

Fairness. The partial results of the election can only be deduced if the cast ballots are decrypted, or the plaintext votes are leaked by the voter's devices. In both of these cases, however, vote privacy would be violated, since the ballots are attached to the voter's identities up until the tallying. Hence, fairness is ensured as long as vote privacy is ensured, namely, under the assumptions (A-BV-HalfVotersHonest, A-BV-NoCoercion, A-BV-CompRestricted, A-BV-VotDeviceTrusted) as shown above.

[8] If only one voter is honest, then the public will not know the correspondences between the voter's identity and the vote; however, if all the other voters are dishonest, and each dishonest voter i reveals the correspondences between the ciphertexts in lists C_{i-1} and C_i to the public, the honest voter will be the one who knows how each one has voted. Thus, vote privacy during anonymization with mix net could be ensured only if at least two voters perform their shuffling correctly and do not reveal the correspondences between the ciphertexts.

Eligibility. Since the identities of the voters whose votes are included in the tally result are public, breaking eligibility would be possible either in case when some of these identities do not belong to eligible voters, or when an eligible honest voter is being impersonated. In the first case, such a manipulation would be evident due to (A-BV-EligVotersKnown). In the second case, as shown above, voter impersonation is unfeasible under the assumptions (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted). In case of homomorphic tallying, the voter might attempt to double-vote by sending an invalid voting option instead of her vote (e.g. an encryption of g^2 with g^1 signifying the "yes"-vote). This, however, is prevented by the proofs of 1-out-of-L encryption (A-BV-CompRestricted). Hence, eligibility is ensured under the assumptions (A-BV-EligVotersKnown), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted).

Vote Integrity. We consider the cases, where a violation of vote integrity would be noticeable by at least one honest voter. Note, in case a majority of the voters is dishonest, then a consensus about the election result might not be reached, since the consistency of the communication can no longer be ensured, as mentioned in the discussion of fault handling. However, we do not consider this to be a violation of vote integrity.

While a malicious voting device can change the voter's vote by encrypting another voting option, this is prevented given the assumption (A-BV-VotDeviceTrusted).

The vote integrity of the election would be broken if a given cast ballot is either dropped from the list of cast ballots, replaced with a ciphertext encrypting another plaintext, or its plaintext content is changed upon decryption. Since the voter is involved in the tallying process, she would notice if her cast ballot is dropped before the tallying. Furthermore, she would also notice if her cast ballot is replaced by an adversary prior to tallying. Alternatively, in case the homomorhic tallying approach is used, another malicious voter might attempt to cast a ballot with negative vote, thus cancelling out some of the other cast ballots (e.g. an encryption of g^{-1} with g^1 signifying the "yes"-vote and g^0 signifying the "no"-vote). This is prevented by the proofs of 1-out-of-L encryption (A-BV-CompRestricted).

During the tallying, in case of mix net approach to the anonymization, the proof of shuffle validity ensures that the contents of the ciphertexts are not modified during anonymization (A-BV-CompRestricted). In case of homomorphic tallying, the proofs of 1-out-of-L encryption submitted during voting ensure that only one the final result represent the sum of cast valid voting options (i.e. that no over-voting or negative voting occurred). The proofs of decryption validity further ensure that the correct plaintext value is being output for each of the cast ballot (A-BV-CompRestricted).

It follows, that vote integrity is ensured in our extension under the assumptions (A-BV-VotDeviceTrusted) and (A-BV-CompRestricted).

Robustness. As mentioned in the discussion of fault handling, the inconsistency of the communication can also hinder the computation of the election result. However, the

Byzantine agreement ensures that the communication is consistent as long as the majority of the voters are honest, their devices are trustworthy and can communicate with each other, and the adversary cannot impersonate honest voters, (assumptions (A-BV-HalfVotersHonest), (A-BV-Communication), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted)). The election key generation ensures that the result of the voting can be decrypted and thus tallied, if at least $\lfloor N/2 \rfloor + 1$ voters and their devices are available and can communicate with each other during decryption (assumptions (A-BV-HalfVotersHonest), (A-BV-Communication) and (A-BV-VotDeviceTrusted). Additionally, the result cannot be tallied without necessarily breaking vote privacy, if the anonymization of the ballots has not been performed correctly, which is possible, as described above, if all but one voter are unable to shuffle the ciphertexts and keep the correspondences between the input list and the shuffled list secret (prevented by the assumptions (A-BV-HalfVotersHonest) and (A-BV-VotDeviceTrusted). Therefore, according to assumptions (A-BV-HalfVotersHonest), (A-BV-Communication), (A-BV-CompRestricted) and (A-BV-VotDeviceTrusted), robustness of the scheme is ensured.

3.5 Related Work

A number of proposals on Internet voting considered elections in boardroom voting setting.

The first proposals for a decentralized election was made by Demillo et al. in [DLM82], implemented in [M$^+$10] and later extended in [AKV05] with regards to an improvement in efficiency. The method used in both these proposals relies on so-called decryption or onion mix net. As opposed to the re-encryption mix net described in Section 2.2.12, in decryption mix net the initial messages are encrypted with the public key of each mix node. Then, each mix node permutes and decrypts all the ciphertexts in its turn. Hence, if even one mix node fails to decrypt, the shuffling cannot be conducted and the initial messages cannot be reconstructed. As in our scheme, the voters in [DLM82] and in [AKV05] act as mix nodes. Hence, due to the usage of a decryption mix net, the schemes in [DLM82, AKV05] are vulnerable with regards to robustness. Namely, in case even one voter fails to provide valid output after casting her ballot, as opposed to our scheme, the result cannot be computed without repeating the voting.

Kiayas et al. [KY02] proposed another approach to boardroom voting, introducing the idea of self-tallying. The self-tallying property ensures that the election result can be tallied directly after the last ballot is cast. For this purpose, the ballots are encoded into the so-called self-dissolved commitments, so that the product of these commitments from all the voters reveals the election result. This approach was further used in several other works. As such, the proposals of [Gro04, HRZ10] improved the efficiency of [KY02]. The proposal in [HRZ10], in particular, was further extended in [KSRH12] in order to improve robustness and in [GIR16] in order to reduce round complexity for multiple elections among the same group of voters. The scheme in [HRZ10] was implemented [MTM16] using the

Ethereum blockchain network [Woo14].

As opposed to our scheme, the proposal by [HRZ10] and its extension in [GIR16] did not ensure robustness, since if even one voter fails to cast her ballot, the final result cannot be computed. Although this vulnerability was remedied in the extension proposed in [KSRH12] by introducing a so-called recovery round performed after voting, it still did not provide robustness in a sense that our proposal does. Namely, in [KSRH12], the tally only includes the ballots by the voters who participate in the recovery round, if such a round is needed. Thus, if some voters fail to send the necessary data in the recovery round after casting their own ballot, the ballots that they cast during voting will not be included. A similar approach is used in [KY02, Gro04] in order to recover the election result in case some of the voters do not cast their ballot. Same as in [HRZ10, GIR16], if some voters fail to send the necessary data during recovery, their ballots that were cast in the previous phase of the election will not be counted. In our proposal, on the other hand, all the cast ballots will be included in the result, even if some the voters are not available afterwards (e.g. due to network problems). Furthermore, all the proposals in [Gro04, HRZ10, KSRH12, GIR16, SP15] rely on an existing PKI for ensuring eligibility, as opposed to our scheme. Finally, they all reveal the election result as the sum of all cast votes. In this way, the votes should be encoded according to the homomorphic tallying approach (Section 2.2.9). This allows to conduct only the elections with simple ballots, such as "yes/no" elections or election with a small number of voting options, of which only one can be selected. With more complex ballots, such as ballots with a large number of voting options or the possibility to rank the voting options, using homomorphic tallying is either too inefficient or impossible (e.g. in case of write-in ballots). Our scheme, on the other hand, allows to conduct elections using the mix net approach for anonymizing the ballots, which supports any kind of ballot complexity including write-in ballots.

The idea of using distributed threshold decryption, similar to our proposal, was used to implement an Android app for boardroom voting in [Rit14]. This app uses a decentralized version of the scheme described in [CGS97][9]. The resulting implementation, however, is vulnerable to an adversary that controls the network (as man-in-the-middle). Hence, such an adversary is able to intercept the messages from the voters, and with that, to violate both vote privacy and vote integrity. Our scheme, on the other hand, is not vulnerable to such attacks due to the decentralized key exchange that allows authenticating and encrypting the messages sent between the voters.

Other approaches to boardroom voting aim to implement boardroom voting schemes without relying on cryptographic techniques. As such, an implementation of boardroom voting system was described and evaluated in [ACW13]. However, as opposed to other boardroom voting schemes, it requires a central trusted instance by using a ballot box which is trusted not to break vote privacy. Another example is an Android application

[9] Note that the election key generation and decryption in Helios-Base also rely on the scheme in [CGS97] as suggested in [CGGI13]

for spontaneous decentralized voting in classroom setting that was proposed in [Esp08]. This proposal, however, relies on a central voting server that is trusted not to manipulate the result. Other boardroom voting schemes, on the other hand, allow to verify that the vote integrity of the election has not been violated.

3.6 Summary and Future Work

The contribution in this chapter extends Helios-Base towards the setting of boardroom voting. As such, this extension allows conducting secure ad-hoc elections without relying on a centralized infrastructure that is required for Helios.

3.6.1 Summary

In order to ensure both the ad-hoc nature of the elections in boardroom voting setting and their decentralization, while preserving the security requirements ensured in Helios, our extension distributes the tasks, previously performed by the trustees in Helios-Base, among the voters while requiring limited preparations. Namely, we relied on such cryptographic primitives as decentralized key exchange for enabling authenticated communication between the voters in absence of a centralized registration authority and public key infrastructure, decentralized communication via Byzantine agreement for ensuring the consistency of the communication in absence of a centralized bulletin board, and distributed key exchange and distributed decryption for tallying the votes in absence of external tabulation tellers.

3.6.2 Future Work

As future work, one would address formally proving the security of the extension. The current definitions for the security requirements such as vote privacy or vote integrity, used to evaluate the security of Helios, are suited towards a central infrastructure and a separation between the voters and the components of the voting system such as the bulletin board or the tabulation teller. Hence, in order to formally evaluate the security of a scheme in boardroom voting, new definitions should be developed that take the decentralized infrastructure and distribution of trust among the voters into account. For this purpose, the literature on secure multi-party computation (e. g. [Gol98]) can be consulted.

Further directions of future work would focus on improving the efficiency of the scheme, considering that the Byzantine agreement requires a high number of communication rounds in order to establish reliable communication channels. Furthermore, one would explore the usability of a scheme, if it is to be implemented. While usability is important for Internet voting in general, it becomes even more crucial in boardroom voting, since all the

tasks within the election now have to be performed by the voters themselves who might not necessarily have a technical background.

Chapter 4

New Voting Setting: Proxy Voting

In well-established forms of elections, the voters express their opinion by voting directly, either for a candidate they want to see as their representative (in representative democracy) or for a particular voting issue (direct democracy). Lately, another form of democracy has been proposed, that provides voters with an additional option: during the election, the voter has the right to either vote herself or delegate her voting right to someone else, such as a trusted expert who might be a public person as well as a trusted friend or relative. Thereby, voters individually have the possibility to decide to which extent they want to directly participate in democratic processes. We refer to the elections in such setting as *proxy voting*.

Our contribution in this chapter is to extend Helios-Base towards proxy voting. For this purpose we also identify the security and functional requirements relevant for the proxy voting settings, on which we base our extension.

This chapter is structured as follows. In Section 4.1, the functional and security requirements that follow from the proxy voting setting are described. Section 4.2 describes the security model that should be ensured in the proposed extension towards proxy voting. The extension is described in Section 4.3 and its security is evaluated in Section 4.4. The related work on proxy voting schemes and implemented software products is overviewed in Section 4.5. The contents of the chapter are summarized and the directions of future work are outlined in Section 4.6.

The contents of this chapter have been published at the *11th International Conference on Availability, Reliability and Security* [KMNV16].

4.1 Setting-Specific Requirements

In this section we describe the proxy voting specific requirements that follow from the proxy voting functionality. Before we list the functional requirements, followed by the security requirements, we start with some pre-considerations. Namely, we consider proxies to be persons that are registered in the voting system, so that the voters could delegate their voting right to the proxies. In this work we do not consider the process of choosing

proxies, since it has no influence on the scheme.

4.1.1 Functional Requirements

There is no established set of requirements in the literature that are considered essential for the proxy voting. Thus, we consider the following functional requirements in this thesis.

Delegation. The voter should be able to choose a proxy from a list of available proxies and transfer her voting right in the election to this person. Then the proxy has the right to vote on behalf of this voter by casting a delegated ballot. Note, that we do not place any restrictions on how and whether the proxy should use her delegated voting right: she can vote for any voting option on behalf of the voter, or not vote at all.

Cancelling the Delegation. After delegating her voting right, the voter should have the option to change her mind and vote herself. Cancelling the delegation should remain possible at any moment of the election prior to the tallying. Note that we assume that the voter's own vote always has the highest priority. Thus we do not account for a scenario whereby the voter casts a direct ballot, but changes her mind and wants to delegate later on.

Alternatively, the voter might be willing to choose different proxies for her delegation. The reasons for this could be twofold. First, in this way the voter can change her mind, if after the delegation she decides to delegate to a different person. In a second use case, the voter wants to delegate to a particular person, but is not sure whether this person would actually use the delegated voting right and cast her delegated ballot. Hence, the voter appoints another proxy who has a lower priority than her first choice. Thus we define following requirements:

Changing the Delegation. After delegating, the voter should be able to appoint a different proxy if she changes her mind.

Prioritizing the Delegation. The voter should be able to assign priorities to different proxies upon delegation. In this case, only the vote from the proxy with the highest priority must be included in the final tally.

4.1.2 Security Requirements

Similar to functional requirements, there are no established list of security requirements for proxy voting in the literature. Hence, we consider following security requirements

that are specific for the delegation process in particular, which we base upon the security requirements for non-delegating voters in Helios-Base. Similar to the general security requirements, we only aim to ensure the delegation-related requirements for the proxies who are not under complete adversarial control.

Secrecy-Related Requirements

As the identity of the proxy chosen by the voter can reveal significant information regarding her political preferences, we consider it important to keep this information private. Hence, we aim to ensure following requirements:

Delegation Privacy - Public. For delegating voters, the identity of the corresponding proxy should not be leaked.

Delegation Privacy - Proxy. For a given delegating voter, a proxy should be unable to tell whether this voter delegated to her or to someone else.

At the same time, as the proxies also participate as voters in the election, we consider it to be important that the principle of secret elections extends to them as well. Note, that there are approaches to proxy voting that suggest the opposite, namely, that the votes of the proxies should be public for the sake of better transparency. However, we consider such an approach to be less optimal, especially in situations where the role of a proxy for a particular voter can be taken not just by a public person, but also by a trusted friend or relative. As such, we include the following requirement:

Vote Privacy for Proxies. The voting system should not provide any information to establish a link between the proxy and her vote, aside from what is available from the election result.

Another challenge refers to a proxy who accumulated a lot of *delegation power* – that is, received a significant number of delegations. This constellation is not inherently problematic, but it might lead to a misuse of power including proxy coercion, if the number of accumulated delegations for each proxy is known to the public or to a third party. Thus, we require that the delegation power of a proxy remains secret to the public. Moreover, a proxy should be restricted in her capability to prove how many votes have been delegated to her, regardless from whether she herself knows this number.

Delegation Power Privacy. The voting system should not reveal any information about the delegation power of the proxy, aside from what is available from the election result. Furthermore, the proxy should be unable to prove both before and after the tallying, how many votes have been delegated to her.

Integrity-Related Requirements.

In order to preserve the integrity of the election, it is further important to ensure, that only the authorised delegations by eligible voters are included in the tallying result, and that they are tallied correctly. Hence, the following three requirements are relevant:

Delegation Eligibility. The proxy should only be able to cast their delegated ballots on the behalf of eligible voters, and at most one delegated ballot per one voter should be included in the final tally.

Delegation Integrity for Voters. No proxy should be able to cast a delegated ballot on the voter's behalf unless authorised by the voter.

Delegation Integrity for Proxies. The valid ballots cast by proxies should be correctly included in the final tally.

4.2 Security Model

In this section we list the assumptions under which the security requirements must be satisfied in our scheme. These assumptions are as follows:

Vote Privacy. Vote privacy should be ensured under the assumptions, that a majority of tabulation tellers are honest, the voting devices of the honest voters do not do not leak voters chosen voting options to an adversary, the bulletin board does not remove or modify the data published on it, the bulletin board shows the same contents to everyone, the voter verifies that she communicates with an authentic bulletin board while casting the vote, the adversary is computationally restricted and the adversary is not capable of coercing the voters to vote for a specific voting option.

Fairness. Fairness should be ensured under the same assumptions as vote privacy, with the assumptions regarding the voters side (the voting devices of the honest voters do not do not leak voters chosen voting options to an adversary, the voter verifies that she communicates with an authentic bulletin board while casting the vote and the adversary does not coerce voters) extending towards proxies and their voting devices as well.

Participation Privacy. Similar to Helios-Base, we do not aim to ensure participation privacy in our extension.

Eligibility. Eligibility should be preserved under the assumptions, that the register of eligible voters is trustworthy, the adversary is computationally restricted and that the voters private signature keys are not leaked by the voting devices.

Vote Integrity. Integrity should be ensured under the assumptions, that the adversary is computationally restrictive, the voters perform the verifications available to them, the bulletin board shows the same contents to everyone and that either the voting devices or the verification devices of the voters are trustworthy.

Robustness. Robustness should be ensured under the assumptions, that more than half of the tabulation tellers are available and provide valid output throughout the election, and that the contents of the bulletin board are available for tally (i.e. the bulletin board does not remove the data published on it and shows the same contents to everyone).

Delegation Privacy. Delegation privacy should be ensured given the assumptions that the channels between the voters and the proxies are private and anonymous, more than half of the tabulation tellers are honest, the bulletin board does not alter the data published on it and shows the same contents to everyone, the voting devices of the voters do not leak information to the adversary, the voters perform the verifications available to them, the adversary is computationally restricted, the voters do not attempt to prove that they delegated to a specific proxy and the proxies are semi-honest (i. e. they follow the delegation protocol without violations, yet might try to gain additional information from the data they receive during the protocol execution).

Vote Privacy for Proxies. Vote privacy for proxies should be ensured under the same assumption as vote privacy, with the assumptions regarding the voters side extended towards proxies. Furthermore, under the assumption that the communication channels between the proxies and the bulletin board are anonymous, vote privacy for proxies should be ensured under only the following additional assumptions: the voting devices of the proxies are trustworthy and the proxies are not coerced to reveal their vote.

Delegation Power Privacy. Delegation power privacy should be ensured under the assumption, that the communication channels between the voters and the proxies are anonymous and private, the communication channels between the proxies and the bulletin board are anonymous, more than half of the tabulation tellers are honest, the bulletin board does not remove or alter the data published on it and shows the same contents to everyone, the voting devices of the voters do not leak information to the adversary, the voters perform the verifications available to them and the adversary is computationally restricted.

Delegation Eligibility. Delegation eligibility should be ensured under the assumptions that the voting register is trustworthy, the voters perform the verifications available to them and the bulletin board does not remove the published data and outputs the same contents to everyone.

Delegation Integrity for Voters. Delegation integrity should be ensured under the assumptions, that the adversary is computationally restricted, the voting devices do not leak secret information and that the channels between the voters and the channels between the voters and the proxies are private and authenticated.

Delegation Integrity for Proxies. Delegation integrity for proxies must be ensured under the assumption, that the proxies perform the verifications available to them, that the bulletin board shows the same contents to everyone, that the voting devices of the proxies are trustworthy and that the adversary is computationally restricted.

The assumptions required for the security of the scheme can hence be summarized as follows:

(A-PV-PrivChannels) The channels between the honest voters and the proxies are private and authenticated.

(A-PV-AnonChannels) The channels between the honest voters and the proxies, as well as between the proxies and the bulletin board, are anonymous.

(A-PV-ProxySemiHonest) The proxies are semi-honest, meaning that they do not deviate from the protocol.

(A-PV-TabTellerHonest) More than half of tabulation tellers are honest and capable of communicating with each other and the bulletin board.

(A-PV-VotDeviceLeakage) The voting devices of both voters and proxies do not leak information to an adversary.

(A-PV-NoBBModification) The bulletin board does not remove or modify the data that is published on it.

(A-PV-BBConsistency) The bulletin board shows the same contents to everyone.

(A-PV-NoCoercion) No coercion or vote selling takes place.

(A-PV-CompRestricted) The adversary is computationally restricted.

(A-PV-Verify) The voters and the proxies perform the verifications available to them within the system.

(A-PV-VerDeviceTrusted) The verification devices of both voters and proxies are trustworthy.

(A-PV-VotRegister) The voting register with the eligible voters public signature keys is trustworthy. Same as in Helios-Base, the assumptions (A-PV-NoBBModification),

(A-PV-BBConsistency) and (A-PV-Verify) are required to ensure the trustworthiness of the voting register, if the voters public signature keys have been generated for a specific election and are only available on the bulletin board.

The assumptions (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-NoBB-Modification), (A-PV-BBConsistency), (A-PV-NoCoercion), (A-PV-CompRestricted), (A-PV-Verify), (A-PV-VerDeviceTrusted) and (A-PV-VotRegister) are the same as in Helios-Base. The assumptions specific to the proxy voting setting, namely, (A-PV-PrivChannels) and (A-PV-AnonChannels), can be explained as follows:

(A-PV-PrivChannels) The exchange of private information between the voter and her chosen proxy is crucial in ensuring that only an authorised proxy can cast a delegated ballot on someone's behalf. Provided a reliable PKI that encompasses the proxies (which can be based upon the same PKI as the voting register), the assumption (A-PV-PrivChannels) can be ensured via end-to-end encryption.

(A-PV-AnonChannels) Unless the anonymity of the communication between voters and proxies is ensured, delegation privacy is broken, as the adversary is able to find out the identity of the voter communicating with a specific proxy. This assumption is furthermore required for delegation power privacy, as the adversary could otherwise find out a lower bound on the delegation power of the proxy, if she observes how many times the proxy cast a delegated ballot. The assumption can be facilitated either by introducing a trusted forwarding server that ensures anonymous communication, or by using onion routing [DMS04].

(A-PV-ProxySemiHonest) Similar to the security model in Helios-Base which does not consider voters who might deviate from protocol (for example, to prove to the adversary how they voted because of coercion), we consider dishonest proxies to be out of scope for our work in this extension. As such, although a dishonest proxy might coerce a voter who delegates to her to reveal the voter's identity thus breaking delegation privacy, we consider such a scenario to be excluded due to the assumption (A-PV-NoCoercion).

4.3 Description

In this section we show how to extend Helios-Base towards proxy voting.

The basic idea of our extension is to introduce a new type of credential, the so-called *delegation credentials*. These credentials $h_{i,j}$ are generated once for each voter, and can be reused in the subsequent elections. The delegation credentials are used by the voters to construct the *delegation tokens* (σ, m, c, π_d) for delegating their vote in each individual election. To do this, tokens are being forwarded to the proxies in an anonymized way and then submitted by the proxies together with the delegated ballots during the voting.

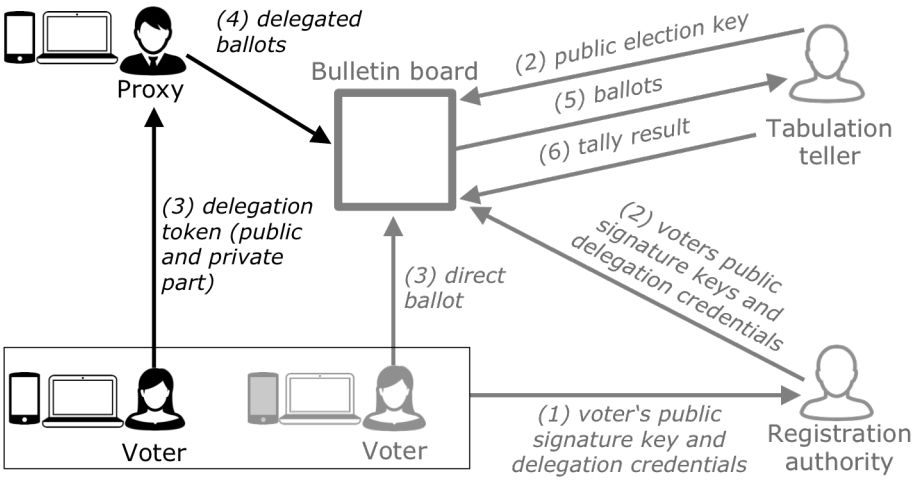


Figure 4.1: Components and their interactions of Helios-PV. The new components compared to Helios-Base are in black. For the sake of simplicity, the verification step using the verification device is omitted, and sending the voter's public signature key to the registration authority can be skipped in case of an existing PKI.

The validity of the tokens, and thus the validity of the delegations, is being verified only in the final stage of the election after further anonymization. In this way, a proxy does not know whether the token she received is valid and whether the delegated ballot of the proxy is included in the tally. If the voter decides to cancel the delegation and cast her vote directly, all her delegation tokens are marked as invalid. In this way, the delegated votes are discarded from tallying.

The components of our proxy voting extension and the interactions between them is depicted on Figure 4.1.

Setup. The initial setup is performed analogously to Helios-Base. The tabulation tellers jointly generate the election key via distributed threshold secret sharing with a threshold of $t = \lfloor N_t/2 \rfloor + 1$ for N_t as the number of tabulation tellers (Section 2.2.11) and publish the public part of it together with the data required for the proofs of the decryption validity. The tabulation teller further verify, that the bulletin board publishes their data correctly. If the PKI with the public signature key from the voters has to be established specifically for the election, the registration authority publishes the public signature keys of the voters, and the voters verify that these public signature keys have been published correctly on the bulletin board. Given a pre-existing trustworthy PKI, the registration authority uses it to publish the list of public signature keys of the eligible voters on the bulletin board, and everyone who has access to the PKI can verify that this list is correct. Furthermore, the voters submit their T public *delegation credentials*, represented as an ordered tuple $h_{i,1}, ..., h_{i,T}$, whereby the voter knows the secret keys $x_{i,j} = \log_g h_{i,j}$, to the registration authority. These credentials are posted by the registration authority on the bulletin board,

and the voters further verify, that the delegation credentials posted next to their identity are valid. For the purpose of establishing the communication channels between the voters and the proxies, the list of available proxies $D_1, ..., D_n$ and their public signature keys is made available as well. Unless specified otherwise, we imply that everything that the voters publish on the bulletin board is digitally signed by their corresponding private signature key. We furthermore imply, that both the voters and proxies verify that they are communicating with an authentic bulletin board during casting a direct ballot or delegating (voters), casting a delegated ballot (proxies) or verifying that the cast ballot, direct or delegated, was properly stored on the bulletin board (voters and proxies). Finally, the list of the valid voting options $\{v_0, ..., v_{L-1}\}$ is published. Note that as in Helios-Base, for the sake of simplicity we describe the election with $L = 2$, however, a generalization towards more complex ballots is possible.

Voting. The voting is the same as in Helios-Base. The voter submits her ballot of the form (c_v, π_v) with c_v as an encryption of the chosen voting option with the public election key and π_v as the proof of well-formedness, which consists, as in Helios-Base either of the proof of plaintext knowledge (Section 2.2.5), or in case of homomorphic tallying approach, also of the proof of 1-out-of-L encryption (Section 2.2.6). The ballot is then published on the bulletin board. For ensuring that her ballot encrypts the correct voting option, the voter can also choose to verify her ballot using the verification device instead of casting it. After submitting the ballot, the voter verifies whether it is correctly published on the bulletin board. Note, that as in Helios-Base, it is crucial for the voters to verify that they are interacting with an authentic bulletin board.

Delegating. For delegating with priority $j = 1, ..., T$, the voter id_i computes an encryption of her credential $c_d = (a_d, b_d) = (g^{r_d}, h_{i,j}h^{r_d})$, a commitment $\sigma = g^m$ of a randomly chosen value $m \in \mathbb{Z}_q$ and a non-interactive signature of secret key knowledge (i.e. the value $x_{i,j}$ on σ Section 2.2.3), $\pi_d = PoK\{(r_d, x_{i,j}) : a_d = g^{r_d} \wedge b_d = g^{x_{i,j}}h^{r_d}\}(\sigma)$. The signature of secret key knowledge is constructed using the technique by by Camenisch et al. (Section 2.2.2) and described in Figure 4.2. The *delegation token*, which are the values (σ, m, c, π_d), are then sent to a proxy of the voter's choice over a private anonymous channel.

Casting a Delegated Ballot. The proxy encrypts her chosen voting option as a ciphertext $c_v = (a_v, b_v) = (g^{r_v}, vh^{r_v})$. She further calculates the proof of knowledge $\pi_v = PoK\{r_v, m \in \mathbb{Z}_q : \sigma = g^m \wedge a_v = g^{r_v} \wedge (b_v = v_0h^{r_v} \vee b_v = v_1h^{r_v})\}$, which serves both as a proof of well-formedness[10] for c_v and as a proof of knowledge of a decommitment value m. The proof, constructed using the techniques by Camenisch et al. (Section 2.2.2),

[10] As in Helios-Base, the proof can be simplified by omitting the proof of 1-out-of-L encryption, in case the mix net approach for anonymising the votes is used.

Private input: $m, r \leftarrow_{\$} \mathbb{Z}_q, h_j \in \mathbb{G}_q, x_j = \log_g h_j \in \mathbb{Z}_q$
Public input: $(g, h), c_d = (a_d, b_d) = (g^{r_d}, h_j h^{r_d}) \in \mathbb{G}_q^2, \sigma = g^m \in \mathbb{G}_q$
Proof:
$w_1, w_2 \leftarrow_R \mathbb{Z}_q, t_1 \leftarrow g^{w_1}, t_2 \leftarrow g^{w_2} h^{w_1}$
$e \leftarrow H(\sigma||g||h||a_d||b_d||t_1||t_2), s_1 \leftarrow w_1 - e r_d, s_2 \leftarrow w_2 - e x_j$
$\pi_d \leftarrow (t_1, t_2, s_1, s_2)$
Verification: (Verify(π_d))
$e \leftarrow H(\sigma||g||h||a_d||b_d||t_1||t_2)$
if $a_d^e g^{s_1} = t_1 \wedge b_d^e g^{s_2} h^{s_1} = t_2$
return 1
else
return $\perp$

Figure 4.2: Signature of secret key knowledge knowledge for a delegation token with priority j.

is described in Figure 4.3. She can then choose to either cast or verify the ballot. The verification is the same as in Helios-Base using the Benaloh challenge. If the proxy decides to cast, she submits $(\sigma, c_v, \pi_v, c_d, \pi_d)$ as her ballot over an anonymous channel. Just as the voters, the proxies verify that their ballot is published on the bulletin board after casting it.

Cancelling a Delegation. If the voter decides to cancel the delegation and vote herself, she just casts her own ballot as in the Helios-Base.

Tallying. After the voting is finished and just before the tallying begins, the voters and proxies can verify that all their ballots, direct and delegated, are stored correctly on the bulletin board and thus are included in further tallying. The tallying then proceeds as follows. First, all duplicate ballots and ballots with invalid proofs or signatures are removed. If the election allows vote updating, all but the last ballot out of the direct ballots cast by the same voter, as well as all but the last ballot out of the delegated ballots cast with the same delegation token, are discarded. The remaining direct ballots are then further used to initialize different sets which are required for the tallying process.

Let $V_{own} = \{(c_v, id)_i\}$ be the set of valid ballots which were cast by voters directly with corresponding voter identities. Let $V_d = \{(c_v, c_d)_i\}$ denote the set of valid ballots and delegation tokens which were cast by proxies, and let $H = \{h_{1,1}, ..., h_{1,T}, ...h_{N,1}, ..., h_{N,T}\}$ denote the set of all valid delegation credentials. For processing the delegated ballots, two sets are initialized: a set $V = \{c : \exists(c_v, id) \in V_{own}\}$ representing the ballots that will be included in the tally (at this step, this set consists of the direct ballots only), and

Private input: $m \in \mathbb{Z}_q, r_v \leftarrow\!\!\$\, \mathbb{Z}_q, i \in \{0,1\}, j = \bar{i}$

Public Input: $(g,h), c_d, c_v = (a_v, b_v) \in \mathbb{G}_q^2, \pi_d \in \mathbb{G}_q^2 \times \mathbb{Z}_q^2, \sigma = g^m$

Proof:

$e_j \leftarrow\!\!\$\, \mathbb{Z}_q$

$w_0, w_1, \hat{w}_0, \hat{w}_1 \leftarrow\!\!\$\, \mathbb{Z}_q$

$t_{1,i} \leftarrow g^{w_i}, t_{2,i} \leftarrow h^{w_i}$

$t_{1,j} \leftarrow g^{w_j} a_v^{e_j}, t_{2,j} \leftarrow h^{w_j} (b_v v_j^{-1})^{e_j}$

$\hat{t}_i \leftarrow g^{\hat{w}_i}, \hat{t}_j \leftarrow g^{\hat{w}_j}$

$e \leftarrow H(\sigma||g||h||a_v||b_v||t_{1,0}||t_{2,0}||\hat{t}_0||t_{1,1}||t_{2,1}||\hat{t}_1)$

$e_i \leftarrow e - e_j, s_i \leftarrow w_i - e_i r_v, \hat{s}_i \leftarrow \hat{w}_i - e_i m$

$s_j \leftarrow w_j, \hat{s}_j \leftarrow \hat{w}_j$

$\pi_v \leftarrow (t_{1,0}, t_{2,0}, \hat{t}_0, t_{1,1}, t_{2,1}, \hat{t}_1, s_0, \hat{s}_0, e_0, s_1, \hat{s}_1, e_1)$

Verification:

if $\mathsf{Verify}(\pi_d) = \perp$

return $\perp$

else $e \leftarrow H(\sigma||g||h||a_v||b_v||t_{1,0}||t_{2,0}||\hat{t}_0||t_{1,1}||t_{2,1}||\hat{t}_1)$

if $e_0 + e_1 = e \ \wedge\ \sigma^{e_0} g^{\hat{s}_0} = \hat{t}_0 \ \wedge\ \sigma^{e_1} g^{\hat{s}_1} = \hat{t}_1$

$\wedge\ a_v^{e_j} g^{s_j} = t_{1,j} \ \wedge\ (b_v v_0^{-1})^{e_0} g^{s_0} = t_{2,0} \ \wedge\ (b_v v_1^{-1})^{e_1} g^{s_1} = t_{2,1}$

return 1

else

return $\perp$

Figure 4.3: Proof of valid delegated ballot for an option v_i with delegation token (σ, m, c_d, π_d).

$H_{own} := \{h_{i,j} : \exists(c_v, id_i) \in V_{own}\}$ as the list of all delegation credentials of voters who cast a direct ballot.

Following procedure is being executed: The delegated ballots $(c_v, c_d) \in V_d$ are being processed through the verifiable re-encryption mix net (Section 2.2.12), resulting in an anonymized list of tuples $\{(c'_v, c'_d)\}$. After the anonymization, the values of c'_d are decrypted to reveal the delegation credentials h' used in constructing the delegating tokens. The ballots with $h' \notin H$ are discarded as cast with non-valid delegation tokens. The rest of c'_v is assigned to the corresponding delegation credential $h_{i,j}$ with i denoting the voter id_i, and j the registration priority.

This procedure results in a new set V'_d that consists of the delegated ballots with valid delegation credentials $(c'_v, h_{i,j})$. The delegated ballots that were overwritten either by the voter herself, or by a delegated ballot with the higher priority are discarded. For this, each encrypted vote c'_v from the tuple $(c'_v, h_{i,j}) \in V'_d$ is added to V if and only if following conditions hold:

1. $h_{i,j} \notin H_{own}$, meaning that the delegated ballot is not revoked by the voter via casting a direct ballot;

2. $\forall(c''_v, h_{i,l}) \in V_d : l < j$, meaning that the delegated ballot is not overwritten with a delegation of higher priority. Note that this implies, that the ballots cast for the same voter with the same delegation priority but different delegation tokens are not included into the final tally.

The encrypted votes in V are being tallied as in the Helios-Base: anonymized using either the mix net or homomorphic tallying (Section 2.2.9) approach, and decrypted with distributed threshold decryption (Section 2.2.11) to reveal the final election result.

4.4 Security Evaluation

In this section we evaluate our extension with regards to the security model given in Section 4.2. Note that our arguments rely on the security of individual components in the scheme. We, however, recognize that a formal proof is required in order to ensure, that the integration of the individual components remains secure, which we leave for future work.

Vote Privacy. As voting remains the same as in Helios-Base, no information about the individual votes is leaked at this stage under the condition that Helios-Base is secure: that is, as long as the majority of the tabulation tellers does not divulge their private election key shares to the adversary, the honest voter or her voting device does not divulge private information used for encrypting her vote to the adversary, the adversary is computationally restricted, the voter verifies that she communicates with an authentic bulletin board

at the time of voting and the bulletin board acts does not remove the ballots or the data submitted to it by tabulation tellers and shows the same contents to everyone (assumptions (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-NoCoercion) and (A-PV-Verify)). After the voting, the direct ballots are anonymized together with the delegated ballots from the proxies, which ensures vote privacy as long as this anonymization is performed correctly. Since the procedure of the anonymization does not differ from Helios-Base, vote privacy for non-delegating voters is preserved under further assumptions that at least one mix node is honest (A-PV-TabTellerHonest) if the mix net approach is used, and the adversary is computationally restricted (A-PV-CompRestricted). The ballot copying attacks on vote privacy are furthermore prevented due to the well-formedness proofs (A-PV-CompRestricted).

Hence, vote privacy is preserved in our extension under the assumptions (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-NoCoercion) and (A-PV-Verify).

Fairness. Same as for vote privacy, the partial results for direct ballots are not leaked under the same assumptions as in Helios-Base. Indeed, since the direct ballots are attached to the voter's identities at all time up until the tallying, revealing partial result would result in breaking vote privacy. Hence, fairness for direct ballots are ensured under the same assumptions as vote privacy, namely, (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-NoCoercion) and (A-PV-Verify).

We now consider the possibilities of violating fairness by getting partial results of the delegated ballots cast by proxies. The votes cast by proxies are encrypted until the final anonymization and subsequent decryption. Thus, similar to vote privacy in Helios-Base, unless the adversary is capable of manipulating the voting device (prevented by the assumption (A-PV-VotDeviceLeakage)), corrupting at least t out of N tabulation tellers (prevented by the assumption (A-PV-TabTellerHonest)), breaking the encryption (prevented by the assumption (A-PV-CompRestricted)), or coercing the proxies to reveal private information (prevented by the assumption (A-PV-NoCoercion)), the adversary gets no information about the partial results from the delegated ballots.

Hence, fairness is ensured as long as the same assumptions that are required for vote privacy hold.

Eligibility. As in Helios-Base, the eligibility of the voters is can be violated if the adversary manages to manipulate the voting register, forge the digital signatures of the voters, or get access to their their private signature keys. A malicious voter can furthermore try to cast an additional vote via overvoting, in case the homomorphic tallying approach is used in the election, for which she would have to falsify the proof of 1-out-of-L encryption.

Hence, eligibility is ensured under the assumptions (A-PV-VotRegister), (A-PV-VotDeviceLeakage) and (A-PV-CompRestricted).

Vote Integrity. The voter has the same options as in Helios-Base to perform the verifications for ensuring that her ballot was cast as intended by the voting device and stored as cast on the bulletin board. Hence, as long as the voters perform the necessary verifications (A-PV-Verify), and their verification devices are trustworthy (A-PV-VerDeviceTrusted), it is ensured that the ballots stored on the bulletin board correspond to the voters intentions. While the adversary might attempt to prevent the voter from verifying that her vote has been stored in the voting system by showing her a different version of the bulletin board, this should be prevented due to the assumption (A-PV-BBConsistency). The stored ballots are properly included in the tally as long as the tabulation tellers provide valid output during anonymization and decryption, which is ensured by the soundness of the proofs of knowledge for computationally restricted adversary (A-PV-CompRestricted). In case the homomorphic tallying approach is used, another malicious voter might prevent some ballots from being included in the tally via negative voting, which should be prevented due to the soundness of the proofs of 1-out-of-L encryption (A-PV-CompRestricted).

Hence, vote integrity is ensured under the assumptions (A-PV-Verify), (A-PV-VerDeviceTrusted), (A-PV-CompRestricted) and (A-PV-BBConsistency).

Robustness. Due to the distributed threshold decryption approach, as long as at least t out of N tabulation tellers participate in the tallying process, which is given due to the assumption (A-PV-TabTellerHonest) for $t > N/2$, the ballots on the bulletin board can be tallied. Hence, as long as the contents of the bulletin board and the majority of the tabulation tellers are available (assumptions (A-PV-NoBBModification), (A-PV-BBConsistency) and (A-PV-TabTellerHonest)), the final result can be computed.

Delegation Privacy. Obviously, some information leakage is unavoidable if a given proxy does not have any voters delegating to her, or if a given voter does not appear in the list of delegating voters, either by voting directly herself or abstaining from the election. This should not be considered to be a violation of delegation privacy. Hence, we consider the following expression for delegation privacy for proxies: Given two delegating voters id_1, id_2, and two semi-honest proxies (A-PV-ProxySemiHonest) D_1, D_2 each receiving a delegation token from one of them, D_1 and D_2 should be unable to distinguish between $((id_1, D_1); (id_2, D_2))$ and $((id_1, id_2); (id_2, D_1))$, with (id_i, D_j) denoting the voter id_i delegating to the proxy D_j.

Given the assumption (A-PV-VotDeviceLeakage), only the information that is either public or sent privately to the proxies could be potentially used for breaking delegation privacy. Furthermore, given the assumption (A-PV-NoCoercion), the voters do not provide any additional information that is not part of the scheme, that might assist in revealing their identity. The assumptions (A-PV-PrivChannels) and (A-PV-AnonChannels) prevent

the proxies from using the communication channels for finding out either the identity of the voters who delegated to each of them, or whether id_1 or id_2 communicated with the other proxy.

Consider the proxies D_1, D_2 casting a ballot $(\sigma_i, c_{v,i}, \pi_{v,i}, c_{d,i}, \pi_{d,i})$ with $(\sigma_i, m_i, c_i, \pi_{d,i})$ as the delegation token, $i = 1, 2$. As the proxies are semi-honest, in addition to the data received from the voters, they only have access to the public information. Namely, each proxy further has access to the encrypted credentials $c_{v,1}$, $c_{v,2}$, the published delegating credentials from both voters $h_{1,1}, ...h_{1,T}, h_{2,1}, ..., h_{2,T}$, and the re-encrypted ciphertexts resulting from the mix net shuffle of delegating credentials $(c'_{v,1}, c'_{d,1}), (c'_{v,2}, c'_{d,2})$.

In order to distinguish between a delegation from id_1 or id_2, the proxies D_1, D_2 need to be able to tell,

- whether $c_{v,1}$ and $c_{v,2}$ encrypt $h_{1,k}$ respectively $h_{2,l}$ or vice versa for some $1 \leq l, k \leq T$, or
- whether $(c_{v,1}, c_{d,1})$ and $(c'_{v,1}, c'_{d,1})$ (respectively, $(c_{v,2}, c_{d,2})$ and $(c'_{v,2}, c'_{d,2})$ encrypt the same plaintexts, or
- whether $(c_{v,1}, c_{d,1})$ and $(c'_{v,2}, c'_{d,2})$ (respectively, $(c_{v,2}, c_{d,2})$ and $(c'_{v,2}, c'_{d,2})$ encrypt the same plaintexts.

Unless the proxies have access to the private election key or the randomness used for re-encrypting $(c'_{v,1}, c'_{d,1}), (c'_{v,2}, c'_{d,2})$, the IND-CPA security of the ElGamal encryption scheme and the zero-knowledge property of the proof π_d restrict them from making the distinction. Thus, given the assumptions (A-PV-PrivChannels), (A-PV-AnonChannels), (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-NoCoercion), (A-PV-CompRestricted) and (A-PV-ProxySemiHonest), delegation privacy against proxies is ensured.

In addition to what a semi-honest proxy might attempt in order to violate delegation privacy, a malicious external adversary can furthermore attempt to trick the voter into encrypting their delegation credential using a different ElGamal key from the one provided by the tabulation tellers. This attack, however, is prevented if the voter verifies that she gets the public election key from an authentic bulletin board (A-PV-Verify), the bulletin board does not change the public election key published by the tabulation teller (A-PV-NoBBModification) and shows the same contents to everyone (A-PV-BBConsistency).

Hence, delegation privacy for the proxies as well as for the external adversary is ensured under the assumptions (A-PV-PrivChannels), (A-PV-AnonChannels), (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-NoCoercion), (A-PV-CompRestricted), (A-PV-ProxySemiHonest), (A-PV-NoBBModification) and (A-PV-BBConsistency).

Vote Privacy for Proxies. We first consider the case, where the anonymity of the communication channels between the proxies and the bulletin board is not assumed. As

shown in the evaluation of fairness, the adversary does not learn the contents of delegated ballots until the voting is finished under the assumptions (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-NoCoercion) and (A-PV-Verify). The adversary, however, can still violate vote privacy without violating fairness given following conditions: she manages to learn the identities of the proxies that cast the delegated ballots by observing the communication channel between the bulletin board and the proxies while learning their vote by preventing the ballots from being anonymized correctly. The proper anonymization, however, is ensured given that the majority of the tabulation teller (who also act as mix nodes in case mix net is used for the anonymisation) are honest (A-PV-TabTellerHonest).

Under an additional assumption that the channels between the proxies and the bulletin board are anonymous (A-PV-AnonChannels), the adversary cannot violate vote privacy even if she manages to decrypt the ciphertexts cast with the delegated ballots. However, vote privacy will be violated if the voting devices of the proxies reveal the voting option that the proxy has voted for (prevented by the assumption (A-PV-VotDeviceLeakage)), or if the proxy herself is coerced to reveal her vote to the adversary (prevented by the assumption (A-PV-NoCoercion)).

Hence, vote privacy for proxies is ensured under the assumptions (A-PV-TabTellerHonest), (A-PV-VotDeviceLeakage), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-NoCoercion) and (A-PV-Verify). Furthermore, under the assumption (A-PV-AnonChannels) vote privacy for proxies is ensured as long as the assumptions (A-PV-VotDeviceLeakage, A-PV-NoCoercion) hold.

Delegation Power Privacy. One way for the adversary to violate delegation power would be count the number of delegated ballots cast by the proxy. Even if some of these ballots are cast using invalid delegation credentials, the adversary can still use this number to estimate the proxy's delegation power and to find out its higher bound. However, this would not be possible given anonymous channels between the proxies and the bulletin board (A-PV-AnonChannels).

The proxy herself knows how many delegation tokens she has gotten in the election – however, she is not able to distinguish, whether a given delegation token contains an encryption of a valid delegation credential unless she manages to decrypt c_v (prevented by the assumptions (A-PV-TabTellerHonest), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency) and (A-PV-Verify)) or to get a plaintext value of it either from the coerced voter or the voting device (prevented by the assumptions (A-PV-VotDeviceLeakage) and (A-PV-NoCoercion)). Hence, she does not know exactly how many of these tokens are actually valid delegations. Furthermore, if the proxy forwards her delegation tokens to a third party, unless this third party does not control the communication channels between the proxy and the voters (which would contradict the assumptions (A-PV-PrivChannels) and (A-PV-AnonChannels)), she does not know which ones of the delegation tokens were sent to the proxy, and which were created by the proxy

herself in order to cheat about her delegating power.

Hence, even if the proxy tries to prove to the adversary how much delegation power she has by showing the delegation tokens to the adversary, she can cheat by either creating some of the delegation tokens herself (thus overestimating her delegation power) or omit some of them (underestimating her delegation power). In both of these cases, the adversary has no way to tell before the tally whether the proxy cheats or not.

After the tallying, however, the total number of invalid delegation tokens $\hat{d}$ cast within the election is revealed. In this way, given d delegation tokens that the proxy received or presented to a third party, it can be concluded that at least $d - \hat{d}$ of them are valid. Thus, the requirement of delegation power privacy after the tally is only probabilistically ensured under the assumptions (A-PV-TabTellerHonest), (A-PV-CompRestricted), (A-PV-NoBBModification), (A-PV-BBConsistency), (A-PV-Verify), (A-PV-PrivChannels) and (A-PV-AnonChannels), which can be corrected if a sufficient number of "chaff" fake delegations are added to the tally similar to the suggestion in the Civitas system [CCM08].

Delegation Eligibility. Given the assumption (A-PV-VotRegisterTrustworthy), only eligible voters participate in the election. Furthermore, given that the voters verify that the bulletin board publishes their valid delegation credentials (assumptions (A-PV-Verify), (A-PV-NoBBModification) and (A-PV-BBConsistency), all the published delegation credentials belong to eligible voters. As long as the decryption of the credentials in delegation tokens is performed correctly, which is ensured by the corresponding proofs of knowledge together with the assumption (A-PV-CompRestricted), everyone can verify that only the delegated ballots with valid delegation credentials with at most one delegated ballot from each voter, are included in the tally. Violating delegation eligibility via overvoting in case of homomorphic tallying is furthermore prevented by the proofs of 1-out-of-L encryption.

Hence, delegation eligibility is ensured under the assumptions (A-PV-VotRegister), (A-PV-CompRestricted), (A-PV-Verify), (A-PV-NoBBModification) and (A-PV-BBConsistency).

Delegation Integrity for Voters. Let us consider the case where the proxy is willing to cast a ballot using the credential $h_{i,j}$ on behalf of some voter id_i. For this she needs to calculate the signature of knowledge $\pi_d = SoK\{(r, x_{i,j}) : a = g^r \wedge b = g^{x_{i,k}} h^r\}(\sigma)$, which according to the assumption, that the adversary is computationally restricted (A-PV-CompRestricted) she cannot do without the knowledge of $x_{i,j}$. The same argument holds for a proxy who wants to cast her ballot with a different priority than the one delegated to her. That is, upon getting the delegation token for the credential $h_{i,j}$, she wants to cast a ballot using the credential $h_{i,k}$ for $k \neq j$. Again, given the assumption (A-PV-CompRestricted), she cannot do this without the knowledge of $x_{i,k}$. Hence, unless $x_{i,j}$ is leaked by the malicious voter's device (prevented by the assumption (A-PV-VotDeviceLeakage)), it cannot be used by the adversary for casting a delegated ballot without the voter's explicit authorization.

An adversary might attempt to impersonate a proxy chosen by the voter and get the delegation credentials, in particular, the private value m, instead of the proxy. This is prevented, however, given private and authenticated communication channels between the voters and the proxies (A-PV-PrivChannels).

Finally, if an adversary gets the public part of the delegation token (namely, the values (σ, c, π_d)) before the delegated ballot is published on the bulletin board (for example, by intercepting the channel between the proxy and the bulletin board or by controlling the bulletin board), she can attempt to use it to cast her delegated ballot instead before the proxy does. However, such an attack would be prevented in our scheme. Due to the assumption (A-PV-CompRestricted) and the fact that σ is integrated into π_d, the adversary would need to know the value of m in order to calculate the proof of knowledge π_v. This, however, is prevented given that m is sent to the legitimate proxy over an private channel (A-PV-PrivChannels) and does not leave the proxy's voting device (A-PV-VotDeviceLeakage).

It is worth noting, that the voter can detect a violation of delegation integrity for voters even without relying on the assumptions (A-PV-VotDeviceLeakage) and (A-PV-PrivChannels) if she performs the available verifications at the end of the tally. Namely, as all the used delegation credentials are decrypted after the anonymisation, the voter can see how many times her delegation credentials (and which priorities exactly) have been used in casting a delegated ballot. If this number or the used priority does not correspond to her intention, or a delegation credentials with the wrong priority has been used, she can conclude that the additional ballots were cast by an adversary. However, while manipulations can be detected in this way, at this point of the election there is no way to distinguish between the ballots from a legitimate proxy or from the adversary without violating delegation privacy.

Delegation Integrity for Proxies. Similarly to the votes of non-delegating voters, the votes of proxies would be correctly included in the election result, as long as the proxies perform the verification and check that their ballots are published on the bulletin board at the end of the election (A-PV-Verify), the proofs of 1-out-of-L encryption (in case of homomorphic tallying) are sound (A-PV-CompRestricted) and the mix net and decryption is performed correctly (ensured by corresponding proofs together with the assumption (A-PV-CompRestricted)).

4.5 Related Work

In this section we review the proxy voting schemes and implementations currently available. There are a few proxy voting implementations which are published by different

organizations. Two widely known systems are LiquidFeedback[11] and Adhocracy[12]. However both approaches completely relinquish vote secrecy, since all actions of users are visible to other users of the system at any time.

Furthermore, a number of schemes were made to conduct cryptographic proxy voting [Tch12, ZHT13].

The ballot-copying approach proposed by by Desmedt et al. [DC12] mentioned in Section 2.3.2 ultimately provides a kind of proxy voting functionality, although the authors do not refer to their proposal as proxy voting. The proxies in this proposal also act as regular voters by casting their ballot as in the original Helios. For delegating, the voter can contact the proxy via a two-way anonymous channel. The proxy then provides the voter with a re-encryption of her own ballot and a valid proof of well-formedness for the ballot, which the voter can blind with random values and cast as her own ballot. This scheme, however, does not allow the proxies to change her mind and update her vote after the voters have cast the ballots received from the proxy, without necessarily contacting all the voters again. Furthermore, the proxy has to cast her ballot or at least know what she should vote for before the voter requests a copy of her ballot. Our extension, on the other hand, does not place such limitations on the proxy, in that a proxy can decide to cast her ballot at any time of the election and does not have to communicate with the voter after she receives the delegation token, even if the proxy wants to update her vote. Furthermore, only one-way anonymous channels are required for the communication between the voter and the proxy in our extension.

Another scheme for proxy voting has been proposed by Tchorbadjiiski [Tch12]. The scheme uses hash chain for implementing the delegation functionality and ensures vote privacy for both voters and proxies as well as delegation privacy using blind signatures and an anonymous channel. The scheme, however, requires trusting a single voting system component for ensuring the eligibility of the election without means to verify it, as opposed to our extension. Similarly, a single voting system component in [Tch12] can violate vote privacy or delegation privacy, while our extension allows to distribute the trust between several entities to secure these requirements. As opposed to our extension, the scheme in [Tch12] further does not provide delegation power privacy, in that each proxy knows and can prove to a third party, how many voters delegated to her.

The scheme proposed in [ZHT13] describes two approaches to delegating. In both of these approaches, the proxies publish their votes in plaintext either before or after the election. For delegating in the *server-side* approach, the voter encrypts the name of the chosen proxy and casts it as her ballot. In the *client-side* approach, the voter encrypts the vote cast by her chosen proxy. As opposed to our extension, both of these approaches do not ensure vote privacy for proxies. Furthermore, in the client-side approach, the proxy cannot update her vote if she changes her mind at any time during the election without

[11] `http://liquidfeedback.org/`
[12] `https://adhocracy.de/`

having to contact all the voters, while our extension allows the proxy to do this at any time until the tallying. In the server-side delegation, the number of voters who delegated to each proxy is public information, hence, as opposed to our extension, delegation power privacy is not ensured. Similar to the proposal in [Tch12], both the client-side and the server-side approach in [ZHT13] are vulnerable to single point of failure, so that a single corrupted voting system component can violate vote privacy and eligibility without being detected, as opposed to our proposal.

The approach in [KNM+16] deals with the problem of coercion resistance as it extends the coercion-resistant JCJ/Civitas scheme [CCM08] towards proxy voting. This is done by introducing delegation credentials, constructed in a similar way to the voting credentials in Civitas, and a new kind of entity, the delegation server, that is to be trusted for coercion resistance. The coercion resistance of the extension, however, as in the Civitas scheme, comes at a price of computational complexity and increased effort that is required from the voters, while the extension described in this chapter provides better efficiency and a simpler process of voting and delegating.

4.6 Summary and Future Work

The contribution described in this chapter extends Helios-Base towards a new form of voting, proxy voting. Elections with proxy voting provide the voters with an additional option to delegate their vote to a trusted proxy instead of voting directly. Our extension allows conducting election with proxy voting functionality, in which the voters can delegate, cancel their delegation if they change their mind and decide to vote directly, and delegate their votes to multiple proxies while assigning different priorities to them.

4.6.1 Summary

Our extension aims to ensure the security of the delegation process, while at the same time preserving the level of security provided by Helios for the voters who vote directly. Namely, our extension ensures such requirements as delegation privacy, meaning that the voter's choice of proxy is secret, delegation eligibility, meaning that the proxies can cast the delegated ballots on behalf of eligible voters only, delegated integrity, meaning that the proxy can only cast a delegated ballot if she is authorized to do so by the voter, and delegation power privacy, meaning that the proxy does not know for sure, how many voters have delegated to her. Our extension further extends the security requirements for direct voters to the proxies. As such, it preserves vote privacy and vote integrity for the delegated ballots cast by proxies. The security of our extension is ensured via the so-called delegation credentials that are assigned to the voters. These credentials are used by the voters to construct anonymized delegation tokens, which are forwarded to the chosen proxies. The proxy uses her delegation token to cast a delegated ballot, which is included in the tally only if it was constructed using a delegation credential from an eligible

voter, the voter did not cast a direct ballot, and did not delegate with higher priority to another proxy. For ensuring both privacy and integrity related security requirements of the delegation, we relied on such cryptographic primitives as proofs of knowledge and signatures of knowledge.

4.6.2 Future Work

As in the case of boardroom voting, one important direction of future work would be the formal evaluation of our scheme. The challenge of this task is due to the fact that a number of new security requirements are introduced that are specific to the delegation process, and therefore have not yet been considered in previous research. Hence, new formal security definitions have to be developed and applied to the evaluation of our scheme as well as possibly other proxy voting schemes.

Other directions of future work would consist researching the non-technical aspects of proxy voting. As such, the usability of our scheme, in particular, of the delegating process, can be studied. Furthermore, it would be interesting to study the understandability of the proxy voting concept in general, including the mental models that the voters might have for delegating their vote, and the additional functionality of the delegation that the voters or the election organizers might want to enable.

Chapter 5

New Voting Setting: Proxy Boardroom Voting

In boardroom voting schemes, time and geographical restrictions often prevent absent board members from participating in the election. As such, even if the election supports remote participation, some of the participants might not be available at the time while the voting takes place. Consequently, decisions are often not supported by a required quorum. For such situation, it is worth considering the possibility to delegate one's vote to a trusted board member that is present, a proxy. We refer to such setting as proxy boardroom voting.

Our contribution in this chapter is to further modify the Helios-Base extension towards boardroom voting described in Chapter 3 to enable proxy boardroom voting.

The chapter is structured as follows. We describe the requirements on proxy voting in boardroom setting in Section 5.1, and the security model under which these requirements are to be achieved in Section 5.2. We describe our extension in Section 5.3, followed by the security evaluation in Section 5.4. Section 5.5 summarizes the chapter and provides the directions of future work.

The contents of this chapter are to be published at the *2nd Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2017* [KMNV17].

5.1 Setting-Specific Requirements

In this section we describe the proxy boardroom voting specific requirements.

The scenario for proxy boardroom voting can be described as follows. The voters know in advance, when the election takes place (i. e. the meeting and the agenda for it). Each voter then can make a decision either to participate in a meeting personally (possibly also remotely), or appoint a proxy before the election, chosen among other meeting participants, who casts a ballot on behalf of the voter during the voting.

The main limitation of this scenario, as opposed to proxy voting with centralized infrastructure, is the ability for the voters to delegate their voting rights before the election,

hence, before the setup phase of the election has been conducted. This is important due to the fact, that the voters who participate in the meeting are supposed to take over the role of tabulation tellers in boardroom voting. Hence, it follows that the setup phase, which requires simultaneous participation of all the voters that would act as tabulation tellers, can only be conducted during the meeting itself. Thus, the following requirement follows:

Delegation Prior to Setup. A voter must be able to appoint the proxy and delegate her voting right before the setup preparations for the elections, which require the participation of all the voters acting as tabulation teller, are conducted.

We further aim to preserve the security requirements specific in proxy voting setting as outlined in Chapter 4.

5.2 Security Model

In this section we list the assumptions under which we aim to ensure the security requirements in our scheme. As we consider the case, where the voters that are present within a meeting also act as proxies, we do not include the security requirements of vote privacy for proxies and delegation integrity for proxies in our analysis. Instead, we consider these security requirements a part of vote privacy and vote integrity respectively. The assumptions that each security requirement relies upon are as follows:

Vote Privacy. Vote privacy should be ensured under the assumptions that more than half of the voters present within a meeting are honest, the voting devices of the honest voters are trustworthy, there is a trustworthy public-key infrastructure (PKI) with the eligible voters public signature keys, the adversary is computationally restricted and the adversary is not capable of coercing the voters to vote for a specific voting option.

Fairness. Fairness should be ensured under the same assumptions as vote privacy.

Participation Privacy. As in Helios-Base and in our boardroom voting extension in Chapter 3, the extension presented in this chapter does not ensure participation privacy.

Eligibility. The eligibility requirement must be ensured under the assumptions that an adversary is computationally restricted, there is a trustworthy PKI with the eligible voters public signature keys and that the voter device does not leak the voter's private signature key to the adversary.

Vote Integrity. Vote integrity should be ensured under the assumptions that the devices of the honest voters are trustworthy, and that the adversary is computationally restrictive.

Robustness. Robustness should be ensured under the assumptions that more than half of all the voters present within a meeting are honest, are able to communicate with each other during the whole election, their devices are trustworthy, and the adversary is computationally restricted.

Delegation Privacy. Delegation privacy should be ensured given the assumptions that more than half of the present voters are not corrupted by the adversary, the voting devices of both delegating and present voters do not leak information to the adversary, the adversary is computationally restricted, there is a trustworthy PKI with the eligible voters public signature keys, and the adversary does not coerce the voters to delegate to a specific proxy.

Delegation Eligibility. Delegation eligibility should be ensured under the assumptions that there is a trustworthy PKI with the eligible voters public signature keys, and that the adversary is computationally restricted.

Delegation Integrity. Delegation integrity should be ensured under the assumptions, that there is a trustworthy PKI with the eligible voters public signature keys, the adversary is computationally restricted, and the voting devices of honest voters are trustworthy.

Delegation Power Privacy. Delegation power privacy should be ensured under the assumptions, that more than half of the present voters are honest, the voting devices of both delegating and present voters do not leak information to the adversary, the adversary is computationally restricted, there is a trustworthy PKI with the eligible voters public signature keys, and the voters do not actively try to prove to the adversary how they voted.

The assumptions required for the security of our scheme are hence summarized as follows:

(A-PBV-HalfVotersHonest) Out of N_p present voters, at least $N_p - t + 1$ are honest and do not divulge their private information to the adversary.

(A-PBV-VotDeviceTrusted) The devices of honest voters are trustworthy.

(A-PBV-Communication) At least t of present voters are available, capable to communicate with each other, and produce valid output during the election.

(A-PBV-CompRestricted) The adversary is computationally restricted.

(A-PBV-NoCoercion) No coercion or vote selling takes place.

(A-PBV-PKI) There is a trustworthy PKI with the eligible voters public signature keys.

The assumptions (A-PBV-HalfVotersHonest), (A-PBV-VotDeviceTrusted), (A-PBV-Communication), (A-PBV-CompRestricted) and (A-PBV-NoCoercion) are required for the boardroom voting as described in Chapter 3. The remaining assumption is justified as follows:

(A-PBV-PKI) Since the proposed scenario does not include ad-hoc elections, it would be possible to assume that the voters have time to exchange their public signature keys in advance (either using the decentralized key exchange as described in Chapter 3 or any other suitable approach), thus establishing a PKI encompassing all the eligible voters.

Note that the setting assumes that the proxies also act as tabulation teller, hence, they are involved during the whole election including the tallying process. This, together with an overall small number of voters or proxies, allows for constructing a scheme for delegation that no longer requires anonymous channels between voters and proxies, as opposed to the scheme described in Chapter 4 at the expense of increased interactions between the proxies. Furthermore, since the proxies are also the voters, the private channels between the proxies and the delegating voters can, on the other hand, be implemented given a trustworthy PKI (A-PBV-PKI).

5.3 Description

We are now ready to provide a description of our scheme for proxy voting in boardroom voting setting. As mentioned in Section 5.2, for this scheme, we assume the existence of a trustworthy public-key infrastructure among all eligible voters. Furthermore, the PKI is used to establish private communication channels between the voters, and the Byzantine agreement (Section 2.2.18), as in Chapter 3, is used for broadcasting the messages among proxies. Depending on the ballot complexity, either the mix net approach (Section 2.2.12) or the homomorphic tallying approach (Section 2.2.9) is used for the anonymization. The components and their interactions of our extension (Helios-PBV) are depicted in Figure 5.1.

In further descriptions we imply that every message is signed by its sender id_i with a private signature key sk_{id_i}.

Pre-Election. A list of all the eligible voters $id_1, ..., id_N$ is made available [13], with a list of their public signature keys pk_{id_i} (the corresponding private signature keys sk_{id_i} are possessed only by the voters). Furthermore, each voter broadcasts a pair of keys (g_i, h_i) with $x_i = \log_{g_i} h_i$ known only to the voter id_i. The list of voters that are about to be present at the meeting is known in advance, so that the majority of them are actually present.

[13] This list, for example, could be a list of board members who have a right to participate in the meeting.

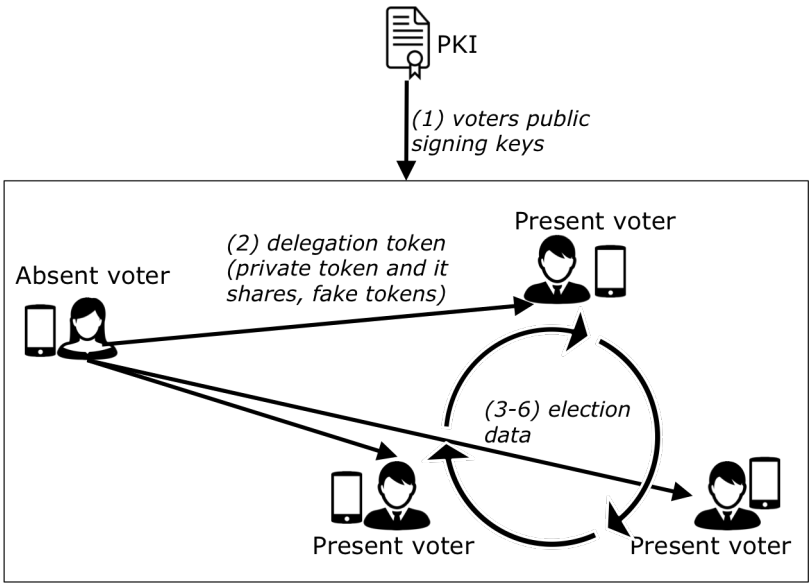


Figure 5.1: Components and their interactions of Helios-PBV. An existing PKI is assumed (1), and after the delegation (2) the proxies jointly conduct the election by exchanging the data in (3) election key generation, (4) casting direct ballots, (5) casting delegated ballots, and (6) tallying.

Delegation. The delegation can occur before as well as during the election, prior to the voting. We define $V_d \subset \{id_1, ..., id_N\}$ as a set of voters who delegate, and $V_p = \{id_1, ..., id_N\} \setminus V_d$ as the voters who decide to vote directly. Since the voters in V_p also receive delegations from the voters in V_d, they are further referred to as proxies.

The threshold t is defined as $\lfloor N_p/2 \rfloor + 1$, with $N_p = |V_p|$ as the number of proxies. If a voter $id_i \in V_d$ decides to delegate, following steps are required:

The voter id_i selects a random value $m_i \in \mathbb{Z}_q$, which serves as her delegation token. She then shares $g_i^{m_i}$ among proxies as follows:

- Compute the shares of m_i using Shamirs secret share scheme (see Section 2.2.10): select a random polynomial $f_i(x) \in \mathbb{Z}_q[x]$ with degree $t-1$ and $f_i(0) = m_i$. For each voter $id_j \in V_p$, compute secret share $m_{i,j} = f_i(j)$.
- For each voter $id_j \in V_p$, furthermore compute commitments $c_{i,j} = (c_{i,j}^{(1)}, c_{i,j}^{(2)})$ with $c_{i,j}^{(1)} = g_i^{r_{i,j}} h_i^{u_{i,j}}$, $c_{i,j}^{(2)} = g_i^{m_{i,j}} h_i^{r_{i,j}}$ for random $r_{i,j}, u_{i,j} \in \mathbb{Z}_q$ (similar to Pedersen commitments, see Section 2.2.13), and a digital signature on $c_{i,j}$, $s_{i,j} = \mathsf{Sign}(\mathsf{sk}_{id_i}, c_{i,j})$.
- For each voter $id_j \in V_p$, set $m'_{i,j}$ to m_i if the voter id_j is chosen as a proxy, and a random value in $\mathbb{Z}_q$ otherwise. If the voter does not want to choose a proxy and wants to abstain instead, she sets $m'_{i,j}$ to a random value in $\mathbb{Z}_q$ for each voter.

The tuple $(g_i^{m_{i,j}}, m'_{i,j}, s_{i,j}, r_{i,j}, u_{i,j})$ is being sent to each voter $id_j \in V_p$ over a private channel. Note that id_j can compute $c_{i,j}^{(1)}, c_{i,j}^{(2)}$ herself.

Setup. At this point, any voter id_i who delegated her voting right can change her mind and attend the meeting; in that case, id_i is excluded from V_d and added to V_p prior to voting.

During the election, the distributed threshold secret sharing (Section 2.2.11) is being executed by the proxies $id_j \in V_p$ to establish the public election key $\mathsf{pk}_v = (g_v, h_v)$ and the corresponding private election key sk_v with $h_v = g_v^{\mathsf{sk}_v}$. At this point the list of valid voting options is being made available, as $\mathbb{V} = \{v_1, ..., v_L\} \subset \mathbb{Z}_q^L$.

Furthermore, for all the delegating voters $id_i \in V_d$ an encryption of the delegation token $g_i^{m_i}$ with pk_v is jointly calculated, whereby each voter $id_j \in V_p$ performs the following steps, given the tuple $(g_i^{m_{i,j}}, m'_{i,j}, c_{i,j}, r_{i,j}, u_{i,j})$ as received during the delegation:

- Encrypt her share of $g_i^{m_i}$ resulting in $e_{i,j}^{(d)} = \mathsf{Enc}(\mathsf{pk}_v, g_i^{m_{i,j}})$,
- Compute the proof of knowledge $\chi_{i,j}$ using the technique described in [CS97b] (see Section 2.2.2), proving that $e_{i,j}^{(d)}$ encrypts the same value that is committed in $c_{i,j}$ (i.e. $\chi_{i,j} = PoK\{r_{i,j}, u_{i,j}, r'_{i,j} : a_{i,j} = g_v^{r'_{i,j}} \wedge b_{i,j}/c_{i,j}^{(2)} = h_v^{r'_{i,j}} h_i^{-r_{i,j}} \wedge c_{i,j}^{(1)} = g_i^{r_{i,j}} h_i^{u_{i,j}}\}$ for $e_{i,j}^{(d)} = (a_{i,j}, b_{i,j})$).
- Broadcast the tuple $(id_i, e_{i,j}^{(d)}, c_{i,j}, s_{i,j}, \chi_{i,j})$.

Given that for each i, at least t of the values of $e_{i,j}^{(d)}$ with valid proofs, $j \in Q_i \subset \{1, ..., N\}$, $|Q_i| \geq t$ are broadcast, these values are combined as

$$e_i^{(d)} = \prod_{j \in Q_i} (e_{i,j}^{(d)})^{\lambda_{i,j}}$$

with $\lambda_{i,j} := \sum_{k \in Q_i, k \neq j} \frac{j}{j-k}$. The resulting value of $e_i^{(d)}$ thus corresponds to the encryption of $g_i^{m_i} = g_i^{\sum_{j \in Q_i} m_{i,j} \lambda_{i,j}}$ with the public election key pk_v.

Voting. The voters who are present in the meeting (i.e. the proxies, $id_j \in V_p$) cast their ballots directly by submitting $E_j^{(p)} = \mathsf{Enc}(\mathsf{pk}_v, v_j)$ with v_j signifying their choice, and the accompanying well-formedness proof σ_j that proves the knowledge of v_j (Section 2.2.5) and, in case of anonymization via homomorphic tallying that $v_j \in \mathbb{V}$ (Section 2.2.6). Furthermore, for each delegating voter $id_i \in V_d$, each proxy $id_j \in V_p$ calculates a value $\hat{e}_{i,j}^{(d)} = \mathsf{Enc}(\mathsf{pk}_v, g_i^{m'_{i,j}})$. Note that $\hat{e}_{i,j}^{(d)}$ encrypts $g_i^{m_i}$ only in case that the voter id_j is in possession of a delegation token m_i (i. e. $m'_{i,j} = m_i$). For the sake of ensuring soundness, the voter further calculates $\pi_{i,j}$ as a proof of knowledge of plaintext discrete logarithm for $m'_{i,j}$ constructed using the technique described in [CS97b] (i.e. $\pi_{i,j} = PoK\{w_{i,j}, m_{i,j} : a_{i,j}^{(d)} = g_v^{w_{i,j}}, b_{i,j}^{(d)} = g_i^{m_{i,j}} h_v^{w_{i,j}}\}$ with $\hat{e}_{i,j}^{(d)} = (a_{i,j}^{(d)}, b_{i,j}^{(d)})$), calculates $E_{i,j}^{(d)}$ as $\mathsf{Enc}(\mathsf{pk}_v, v_{i,j}^{(d)})$ with $v_{i,j}^{(d)}$ as her chosen option to cast on behalf of the delegating voter id_i, $\sigma_{i,j}$ as the well-formedness proof for $E_{i,j}^{(d)}$ (i. e. as for the direct ballots, the proof of plaintext knowledge,

and for homomorphic tallying, the proof of 1-out-of-L encryption), and broadcasts the tuple $(\hat{e}_{i,j}^{(d)}, E_{i,j}^{(d)}, \pi_{i,j}, \sigma_{i,j})$.

Tallying - Weeding Duplicates and Invalid Delegations. In the next stage, the delegated ballots are processed. First, the delegated ballots with invalid proofs of knowledge are removed. Then, the vote updating policy is applied. Namely, the given two ballots cast as direct ballots by the same voter, or two delegated ballots cast on behalf of the same voter by the same proxy, either all but the last (if vote updating is allowed) or all by the first (if vote updating is not allowed) cast ballot are excluded from further processing.

The next step removes the delegated ballots if they have canceled by the voter, i.e. if the voter cast a direct ballot instead. Namely, out of all the delegated ballots tuples $(\hat{e}_{i,j}^{(d)}, E_{i,j}^{(d)}, \pi_{i,j})$, the ballots with $id_i \in V_p$ are removed.

The remaining delegated ballots are being anonymized via verifiable re-encryption mix net (Section 2.2.12) with each proxy acting as a mix node, resulting in an anonymized list $V = \{(\hat{e}_{i,j}^{'(d)}, E_{i,j}^{'(d)})\}$. The values $e_i^{(d)}$ that encrypt the voters delegation tokens m_i are also processed through the mix net resulting in an anonymized list $V' = \{e_i^{'(d)}\}$. The next step removes the delegated ballots cast with an invalid delegation token. For this, the following procedure is performed for each anonymized tuple $(\hat{e}_{i,j}^{'(d)}, E_{i,j}^{'(d)}) \in V$:

- Calculate $\mathsf{PET}(\hat{e}_{i,j}^{'(d)}, e_i^{'(d)})$ for each $e_i^{'(d)} \in V'$ (see Section 2.2.17).
- If the PET is positive for some index i:
  - add $E_{i,j}^{'(d)}$ to the list V'' for further tallying,
  - remove $e_i^{'(d)}$ from V'.

Tallying - Mixing and Decrypting. After that, the list of ciphertexts $\{E_j^{(p)}\}_{id_j \in V_p} \cup \{E_i^{'(d)}\} \in V''$ is being anonymized with either mix net or homomorphic tallying approach. The anonymized result is being decrypted via distributed threshold decryption (Section 2.2.11).

5.4 Security Evaluation

In this section we provide an informal security evaluation for our scheme. Note that, as in previous chapters, we recognize that for better reliability of our arguments a formal proof is required, which we consider to be a part of future work.

We first evaluate the fulfillment of the security requirements not related to delegation.

Vote Privacy. This requirement is violated if the adversary is capable of corrupting voting devices, which then leak the choices made by the voters. This attack is infeasible as-

suming that honest voters' devices are not compromised (A-PBV-VotDeviceTrusted), and the voters are not trying to prove to the adversary how they voted (A-PBV-NoCoercion).

Another way to violate vote privacy of honest voters is to decrypt the encrypted votes prior to their anonymization (i. e. before the mixing or homomorphic multiplication). This, however, requires either being able to break the encryption of the ballots (prevented by the assumption (A-PBV-CompRestricted)), the collaboration of at least t voters who hold the shares of a private election key (prevented by the assumption (A-PBV-HalfVotersHonest)), or eavesdropping on the communication channels in order to get the private election key shares, which is prevented by a trustworthy PKI (A-PBV-PKI).

Furthermore, vote privacy can be violated by revealing the secret permutation used by each voter during the mixing. However, as long as at least one voter keeps this permutation secret (implied by (A-PBV-HalfVotersHonest)), the permutation between the resulting output and the input ciphertexts remains secret to the public, and as long as at least two voters keep their permutation secret (A-PBV-HalfVotersHonest), the the permutation between the resulting output and the input ciphertexts remains secret to all the voters as well.

Hence, vote privacy is ensured under the assumptions (A-PBV-VotDeviceTrusted), (A-PBV-NoCoercion), (A-PBV-PKI), (A-PBV-CompRestricted) and (A-PBV-HalfVotersHonest).

Fairness. As the cast ballots are attached to the voters' identities until the tallying, violating fairness would also imply violating vote privacy. Hence, fairness is ensured under the same assumptions as vote privacy: namely, that the voting devices of honest voters are trustworthy (A-PBV-VotDeviceTrusted), at least half of proxies are honest (A-PBV-HalfVotersHonest), no coercion or vote selling takes place (A-PBV-NoCoercion), the PKI is trustworthy (A-PBV-PKI) and the underlying encryption cannot be broken (A-PBV-CompRestricted).

Eligibility. This requirement is ensured as long as the PKI used to authenticate the voters is trustworthy (A-PBV-PKI), the private signature keys are not leaked to the adversary by the honest voters voting devices (A-PBV-VotDeviceTrusted), and the adversary is computationally restricted (A-PBV-CompRestricted).

Vote Integrity. Similar to the security evaluation in Chapter 3, we consider the cases, where a violation of vote integrity would be noticeable by at least one honest voter. For direct ballots, vote integrity can be violated by replacing a cast ballot with another ciphertext at the time of casting via a malicious voting device (prevented by A-PBV-VotDeviceTrusted). Alternatively, the adversary can try either to replace or to drop the ballot after it has been cast[14]. However, given a trustworthy voting device, such a

[14]Note that these attempts would also be prevented given assumptions (A-PBV-PKI, A-PBV-Communication)

manipulation will be detected by the voter, since her result would not fit with the result of other voters (A-PBV-VotDeviceTrusted). Another way to manipulate the tally would be to manipulate ballots during the shuffling process or to produce an incorrect decryption result. Both possibilities are excluded due to the usage of sound proofs of shuffle validity and decryption validity (A-PBV-CompRestricted).

We now consider the integrity of delegated ballots. Note that in case the voter has delegated her voting right to multiple proxies, only a ballot from one of them is included into the tallying. Hence, in this way excluding the ballots of other proxies from being included in the tallying is not considered a violation of delegation integrity for proxies. Similarly, excluding the ballots cast on behalf of dishonest voters does not violate the requirement.

A dishonest majority of proxies might prevent the delegated ballot on behalf of the particular voter from being included in the tally by refusing to publish their values $e_{i,j}^{(d)}$, hence, preventing the reconstruction of $e_i^{(d)}$. However, since the misbehaviour of dishonest voters would be detected by everyone, we do not consider it to be a violation of vote integrity.

On the other hand, publishing the invalid values $e_{i,j}^{(d)}$, so that the reconstructed $e_i^{(d)}$ does not encrypt the value of $g_i^{m_i}$ for a valid delegation token m_i, would indeed be a violation of vote integrity, if undetected. However, the soundness of the proof of knowledge $\chi_{i,j}$ that accompanies $\hat{e}_{i,j}^{(d)}$ and the computational binding property of the commitment $c_{i,j}$ that holds unless the secret x_i is leaked (assumptions (A-PBV-VotDeviceTrusted) and (A-PBV-CompRestricted)) ensure that each $e_{i,j}^{(d)}$ encrypts the value $g_i^{m_{i,j}}$ committed in $c_{i,j}$. Since $c_{i,j}$ is signed by the voter (and a lack of a valid signature would be noticeable to the honest proxies, as well as to the delegating voters who verify the election data), the unforgeability of the signature (A-PBV-CompRestricted) ensures that $c_{i,j}$ was sent by the voter herself, hence, it contains the valid value of $g_i^{m_{i,j}}$. Furthermore, reusing old signatures on $c_{i,j}$ would be prevented, since the election information and the timestamp are incorporated in the signature. Hence, the reconstructed value $e_i^{(d)}$ encrypts the same $g_i^{m_i}$ that is shared by the voter id_i

Another way to prevent the delegated ballots from an honest proxy to be included in the tally is to ensure that the result of $\mathsf{PET}(\hat{e}_{i,j}^{'(d)}, e_i^{'(d)})$ outputs some value other than 1. This is prevented due to the soundness of the proofs of knowledge accompanying the PETs. Furthermore, analogously to the case of direct ballots, the soundness of proofs of shuffle validity and decryption validity prevent the manipulation of cast ballots (A-PBV-CompRestricted).

It therefore follows that vote integrity is ensured under the assumptions (A-PBV-VotDeviceTrusted) and (A-PBV-CompRestricted).

Robustness. Similar to the scheme in Chapter 3, the consistency of broadcast communication via Byzantine agreement is ensured under the assumptions (A-PBV-HalfVotersHo-

nest), (A-PBV-Communication), (A-PBV-CompRestricted) and A-PBV-VotDeviceTrusted). Furthermore, violating robustness would mean that either the mixing, the weeding of invalid delegations or the decryption has failed to output a valid output. This is prevented if at least t proxies are available and provide the required output during the tallying (A-PBV-Communication).

Delegation Privacy. The delegation privacy requirement would be violated if it is revealed which proxy possesses the value m_i that was shared by the voter id_i among other proxies. This can be achieved either by corrupting the voting device of id_i that stores m_i (prevented by the assumption (A-PBV-VotDeviceTrusted)), observing the voter during the delegation (prevented by the assumption (A-PBV-NoCoercion)), getting access to at least t shares of m_i (i.e. corrupting at least t proxies (prevented by the assumption (A-PBV-HalfVotersHonest)) or their voting devices (prevented by the assumption (A-PBV-VotDeviceTrusted)), or eavesdropping on the communication channels (prevented by the assumption (A-PBV-PKI))), or decrypting $e_i^{(d)}$ and the values of $\hat{e}_{i,j}^{(d)}$ (i.e. either breaking encryption (prevented by the assumption (A-PBV-CompRestricted)) or obtaining at least t shares of a secret key sk_v by corrupting at least t proxies (prevented by the assumption (A-PBV-HalfVotersHonest)) or their voting devices (prevented by the assumption (A-PBV-VotDeviceTrusted))).

Furthermore, the delegating voter herself cannot construct a proof that she delegated to a specific proxy, even if she provides all the shares $g_i^{m_{i,j}}$ and the value of m_i to the adversary. Namely, given that the voter knows the discrete logarithm $x_i = \log_{g_i} h_i$, she can provide fake values of $g_i^{m_{i,j}}$, m_i instead. As such, for every values $m_{i,j}$, $r_{i,j}$ and $u_{i,j}$ (thus, for every pair of commitments $c_{i,j}^{(1)}$, $c_{i,j}^{(2)}$) and every value $m'_{i,j} \neq m_{i,j}$ the voter can find $r'_{i,j}$, $u'_{i,j}$ so that $x_i r_{i,j} + m_{i,j} = x_i r'_{i,j} + m'_{i,j}$ and $x_i u_{i,j} + r_{i,j} = x_i u'_{i,j} + r'_{i,j}$ (thus, $c_{i,j}^{(1)} = g_i^{r'_{i,j}} h_i^{u'_{i,j}}$ and $c_{i,j}^{(2)} = g_i^{m'_{i,j}} h_i^{r'_{i,j}}$). She can then fake the receipt by sending a random value m'_i and a set of shares $m'_{i,j}$ that reconstruct to m'_i together with the corresponding values of $r'_{i,j}, u'_{i,j}$ to the present voter who requests such a receipt. Given t as threshold and N_p as the total amount of present voters among which g^{m_i} is shared, the voter would have to fake at least $N_p - t + 1$ shares $m_{i,j}$. Hence, as long as at least $N_p - t + 1$ present voters are honest, and that the delegating voter knows the identities of the honest present voters, the adversary would not be able to distinguish between the fake values $g_i^{m_{i,j}}$, m_i that from the real ones.

Note, however, that in case one of the voters $id_j \in V_p$ (i.e. who received delegations) is not available in the meeting, the scheme reveals the number of delegating voters who either abstained (but still participated in the delegation by issuing invalid delegation tokens $m'_{i,j} \neq m_i$ to all the voters in V_p) or delegated to id_j. We do not consider such a case to be a violation of delegation privacy, since, as shown above, the scheme does not reveals the identities of the voters who either issued invalid delegation tokens or delegated to id_j and does not make it possible to tell whether a given voter issued a valid token

to id_j or not (under the assumptions (A-PBV-VotDeviceTrusted), (A-PBV-NoCoercion), (A-PBV-HalfVotersHonest), (A-PBV-PKI) and (A-PBV-CompRestricted)). At the same time, in order to reduce the information leakage in our scheme, we would suggest actively encouraging that the voters in V_d who decide to abstain still to participate in the delegation phase of the election by issuing invalid delegation tokens $m'_{i,j} \neq m_i$ to all the voters in V_p. Furthermore, the proxies can be encouraged to re-delegate by forwarding their delegation token to another trusted proxy, if they think they would not be able to participate in the meeting.

Hence, under the assumptions (A-PBV-VotDeviceTrusted), (A-PBV-NoCoercion), (A-PBV-HalfVotersHonest), (A-PBV-PKI) and (A-PBV-CompRestricted), delegation secrecy is ensured.

Delegation Eligibility. Casting a delegated ballot on behalf of a non-eligible voter would require forging the signatures on the commitments $c_{i,j}$ sent to the proxies (prevented by the assumptions (A-PBV-PKI) and (A-PBV-CompRestricted)). Furthermore, multiple delegated ballots on behalf of the same voter are dismissed during tallying.

Hence, this requirement is ensured as long as the PKI is trustworthy (A-PBV-PKI) and the adversary is computationally restricted (A-PBV-CompRestricted).

Delegation Integrity. One way to violate this requirement for a proxy id_j who wants to cast a delegated ballot on behalf of the voter id_i without being authorized, is to find out the value of m_i, shared by id_i to the proxies during the delegation. This would require either corrupting the voting device of id_i (A-PBV-VotDeviceTrusted) or eavesdropping on the communication between id_i and a proxy chosen by her (which is prevented due to private communication channels, i.e. the trustworthiness of the PKI (A-PBV-PKI)). Note that even if the adversary succeeds in obtaining at least t shares of $g_i^{m_{i,j}}$ from the present voters, she would still require to compute the discrete logarithm $m_{i,j}$ (prevented by the assumption (A-PBV-CompRestricted)).

Alternatively, an adversary can attempt manipulating the computation of $e_i^{(d)}$, so that it encrypts a plaintext $g_i m'$ chosen by her. As shown in the evaluation of vote integrity, however, the assumption (A-PBV-CompRestricted) ensures than $e_i^{(d)}$ encrypts the same value $g_i^{m_{i,j}}$ sent by the voter.

Furthermore, delegation integrity can be violated, if the proxy id_j submits a value $\hat{e}_{i,j}^{(d)}$ which is accepted during the weeding of invalid delegations. The soundness of the proof of knowledge of plaintext discrete logarithm $\pi_{i,j}$ (A-PBV-CompRestricted) ensures that the proxy knows the value $m_{i,j}$ with $g_i^{m_{i,j}}$ encrypted in $\hat{e}_{i,j}^{(d)}$. As shown above, the assumptions (A-PBV-PKI) and (A-PBV-CompRestricted) ensure that the reconstructed values $e_i^{(d)}$ encrypt the delegation tokens submitted by the voters to their chosen proxies. The soundness of the proof of shuffle ensures (A-PBV-CompRestricted), that the anonymized encrypted delegation tokens $e_i^{'(d)}$ encrypt the same values as $e_i^{(d)}$ as the reconstructed

values sent by the voters. The soundness of proofs of knowledge accompanying the PET (A-PBV-CompRestricted) ensures that the delegation by id_j on behalf of id_i is accepted only if $\hat{e}_{i,j}^{\prime(d)}$ encrypts the same values as $e_i^{\prime(d)}$. Hence, the proxy is capable of submitting $\hat{e}_{i,j}^{(d)}$ that is accepted as valid only if she knows m_i.

Hence, delegation integrity is ensured under the assumptions (A-PBV-CompRestricted), (A-PBV-PKI) and (A-PBV-VotDeviceTrusted).

Delegation Power Privacy. Given $N_d = N - N_p$ delegating voters, each proxy should posses N_d delegation tokens. Violating delegation power privacy would mean estimating, possibly with the help of the proxy herself who tries to prove her delegation power, how many of those tokens are valid. However, given that the delegation privacy requirement is fulfilled, a proxy herself does not know which ones of the delegation tokens she received are valid. Hence, under the assumptions that at least $N_p - t + 1$ of proxies are honest (A-PBV-HalfVotersHonest), the PKI ensures the privacy of the communication channels (A-PBV-PKI), the voting devices of the delegating voters and honest proxies are trustworthy (A-PBV-VotDeviceTrusted), the voters do not collaborate with the proxy to prove that they delegated their voting right to her (A-PBV-NoCoercion) and the encryption is not broken (A-PBV-CompRestricted), delegation power privacy is ensured.

Note that as already mentioned in the evaluation of delegation privacy, if a proxy $id_j \in V_p$ does not participate in the election, our scheme could reveal the number of voters N_j who either delegated to id_j or issued invalid delegation tokens $m_{i,j}$ to all the proxies. However, since the scheme does not reveal, how many voters out of N_j abstained, delegation power privacy is not violated, especially if the voters who want to abstain are encouraged to issue invalid delegation tokens instead of not participating at all.

5.5 Summary and Future Work

The contribution described in this chapter extends Helios towards elections in the proxy boardroom setting, distributing the trust between the voters during the election in a decentralized way, and allowing the voters who cannot be available during the election to delegate their vote to one of the present voters before the election starts.

5.5.1 Summary

Building up upon our extensions described in Chapter 3 and Chapter 4, our scheme ensures the security requirements that are relevant for boardroom voting for direct voters, and security requirements relevant for the delegation process in the proxy voting setting. The extension employs such cryptographic primitives as secret sharing, commitments and proofs of knowledge. In order to delegate their vote before the election setup, the voter generates a delegation token and shares it via secret sharing among the present voters. She further sends the delegation token to the present voter who is chosen as a proxy, and

sends fake delegation tokens to the rest of the present voters. During the election, each one of the present voters uses the delegation token that she received to cast a delegated ballot, with only the ballot cast with a real delegation token included in the tally.

5.5.2 Future Work

Generally, the same directions of future work that apply to the boardroom voting and proxy voting setting, also apply to proxy boardroom setting. As such, one would provide formal security proofs for the proposed proxy boardroom voting scheme. For this purpose, similar to the challenges for boardroom voting and proxy voting, new formal definitions have to be developed that address the infrastructure and functionality in the proxy boardroom voting setting. Further directions of future work would include studying the usability and understandability of proxy voting concept in boardrooms, and the additional functionality that might be useful in such setting.

Another direction of future work is more specific to the proxy boardroom voting setting. It can be generally presumed that the number of voters in boardroom voting in general is relatively small, as compared to large-scale elections. Hence, due to the small size of the electorate, the election result itself might reveal enough information that allows to identify the votes of individual voters. This issue might become even more prominent if some of the voters delegate, since all the ballots, direct and delegated, are cast by a small pool of present voters. Hence, it becomes more important to study the methods to reduce the information leakage from the election result, for example, by employing tally-hiding methods that reveal only the winner of the election, but not how many votes each voting option has.

Chapter 6

Privacy Improvements: Participation Privacy, Receipt-Freeness

For the sake of improving eligibility, Helios-Base does not provide participation privacy by publishing the identities of the voters who cast their ballots. It is also vulnerable to vote buying by not ensuring receipt-freeness: hence, the voter is able to construct a receipt that proves how she voted.

Our contribution is to propose an extension of Helios-Base, further referred to as KTV-Helios, that ensures probabilistic participation privacy and receipt-freeness while also providing means to verify that only the eligible voters have cast their ballots. We provide formal proofs for the security of our extension. For these proofs, the existing definitions are used for vote privacy, vote integrity and eligibility. The new definitions for fairness as well as for probabilistic participation privacy and probabilistic receipt-freeness are proposed. For participation privacy and receipt-freeness, the means to quantify the adversarial advantage in KTV-Helios according to these definitions have also been described.

This chapter is structured as follows. We describe the additional security requirements as ensured in our scheme in Section 6.1, and the security model we rely on in Section 6.2. In Section 6.3 we provide a formal description of KTV-Helios. Next we provide the formal security proofs for our extension in Section 6.4. Namely, we provide definitions of existing security requirements and prove their fulfillment for KTV-Helios: vote integrity and eligibility in Section 6.4.4, and fairness, vote privacy and its stronger form, receipt-freeness, in Section 6.4.1. In Section 6.4.3 we define participation privacy and provide an evaluation of it for KTV-Helios, and Section 6.4.1 defines and evaluates receipt-freeness. We further informally evaluate robustness in Section 6.4.5. Section 6.5 describes the related work on schemes ensuring the security requirements introduced by our extension and on formal security proofs for electronic voting schemes. Section 6.6 discusses the efficiency of the proposed extension, and Section 6.7 discusses the integration of KTV-Helios extension with other settings described in Chapter 3 (see Section 6.7.1), Chapter 4 (see Section 6.7.2) and Chapter 5 (see Section 6.7.3). The contributions of this chapter are summarized and directions of future work are discussed in Section 6.8.

The contents of this chapter have been published at the *5th International Conference on E-Voting and Identity* [KTV15] and the *Cryptology ePrint Archive* [BKV16].

6.1 Extension-Specific Requirements

In this section we describe the privacy improvements that we introduce to Helios-Base in our extension. One of those improvements is the introduction of participation privacy: Although the information whether a voter has participated in the election is usually potentially available in traditional paper-based elections, whereby anyone can observe people going into a polling station, an Internet voting system without participation privacy reveals the identities of the voters who cast their vote in an election on a much larger scale by publishing them online. Hence, the lack of participation privacy in Internet voting is a violation of voter privacy that is more serious in comparison to paper-based elections.

Participation Privacy. The voting system should not reveal whether a particular voter has participated in the election.

Another additional security property with regards to privacy is meant to prevent vote buying. While in Helios-Base the voter can provide a receipt showing which voting option she voted for by storing the randomness used to encrypt her ballot, we aim to prevent such attacks in our extension. Hence, the next requirement is to ensure the following property:

Receipt-Freeness. Vote privacy should be preserved under the assumption, that the voter might attempt to construct a receipt that proves to the adversary, that the voter has voted for a specific voting option.

We provide the formalisation of both participation privacy and receipt-freeness as used in our extension in Section 6.4.3 and Section 6.4.1.

6.2 Security Model

In this section we describe the assumptions that the proposed security requirements rely on in the KTV-Helios scheme. Our extension introduces a new kind of entity, the *posting trustee*. The security requirements should be fulfilled under following assumptions in our scheme:

Vote Privacy. For the voters who do not attempt to prove how they voted, the vote privacy requirement must be ensured under the assumptions, that a majority of tabulation tellers are honest, the voting device does not leak information to the adversary, the adversary is computationally restricted, the voters verify that they are interacting with the valid bulletin board during voting and the bulletin board does not remove or modify the data

published on it and shows the same contents to everyone. Furthermore, for the voters who might attempt to provide a receipt that proves how they voted, **receipt-freeness** should be ensured under the following additional assumptions: the verification devices do not leak information to the adversary, the adversary is incapable of observing the communication channel between the voter, the posting trustees and the voting system, at least one posting trustee is honest, the voter is capable of casting a vote without being observed by the adversary, the adversary does not cast ballots on behalf of the voter which plaintexts the voter does not know, and the voters decide to follow the adversary's instructions or to fake their receipts independently from each other.

Fairness. Fairness must be ensured for the voters who do not interact with the adversary under the same assumptions as vote privacy without receipt-freeness.

Participation Privacy. Participation privacy must be ensured given the following assumptions: the majority of the tabulation tellers are honest, both the voting and the verification devices do not leak information to the adversary, the adversary is incapable of observing the communication channel between the voter, the posting trustees and the voting system, at least one posting trustee is honest, the voters verify that they are interacting with the valid bulletin board during voting, the bulletin board does not remove or modify the data published on it and shows the same contents to everyone, the voters decide to abstain or to participate in the election independently from each other and the adversary does not coerce the voters to abstain from the election.

Vote Integrity. The vote integrity of the election must be ensured as long as the voters who cast their ballots perform the available verifications, the bulletin board shows the same contents to everyone, the voters' devices do not leak the voters private signature keys to the adversary, and the adversary is computationally restricted.

Eligibility. Eligibility must be ensured under the assumption, that the voting register is trustworthy and the adversary is computationally restricted.

Robustness. Availability should be preserved under the assumptions, that the contents of the bulletin board are available for tallying (i.e. the bulletin board does not remove the data published on it and shows the same contents to everyone), and more than half of the tabulation tellers are honest.

The assumptions used in our scheme can hence be summarized as follows:

(A-KTV-TabTellerHonest) More than half of tabulation tellers are honest.

(A-KTV-PosTrusteeHonest) At least one posting trustee is honest.

(A-KTV-VotDeviceLeakage) The voting devices of voters do not leak information to an adversary.

(A-KTV-NoBBModification) The bulletin board does not remove or modify the data published on it.

(A-KTV-BBConsistency) The bulletin board shows the same contents to everyone.

(A-KTV-CompRestricted) The adversary is computationally restricted.

(A-KTV-Verify) The voters perform the verifications available to them within the system.

(A-KTV-VerDeviceTrusted) The verification devices of voters are trustworthy.

(A-KTV-VotRegister) The voting register with the eligible voters public signature keys is trustworthy. If the voters public keys are only available on the bulletin board, the assumptions (A-KTV-NoBBModification, A-KTV-BBConsistency) are further required.

(A-KTV-AnonChannels) The channels between the honest voters and the bulletin board, as well as between the posting trustees and the bulletin board, are anonymous.

(A-KTV-NoForcedAbstention) No coercion in form of forced abstention or randomization takes place.

(A-KTV-NoUnknownPlaintext) The adversary does not cast ballots on behalf of the voter which plaintexts the voter does not know.

(A-KTV-HiddenVote) The voters have the possibility to cast a ballot without being observed by the adversary.

(A-KTV-IndAbstain) The honest voters make their decision to abstain or to participate in the election independently from each other.

(A-KTV-IndReceipt) The voters who are required by the adversary to provide receipts decide to follow the instructions or to fake the receipt independently from each other.

The assumptions (A-KTV-TabTellerHonest), (A-KTV-VotDeviceLeakage), (A-KTV-NoBBDeletion), (A-KTV-CompRestricted), (A-KTV-Verify), (A-KTV-VerDeviceTrusted) and (A-KTV-VotRegister), are the same as in Helios-Base. Note, however, that the assumption (A-KTV-VotDeviceLeakage) is required not only for vote privacy, as in Helios-Base, but for the vote integrity as well, since it relies on the adversary not having access to the voter's private signature key. This assumption, however, could be realistically expected in some of the settings, e.g. in case a national eID infrastructure with tamper-resistant smartcards for storing the voter's private signature key is in place.

The assumption that the voter's private signature key is not leaked is also crucial for ensuring receipt-freeness. Note, that the voter might still violate it by either voluntarily divulging her private signature key to the adversary or even giving away her smartcard entirely, if she wants to prove how she voted. However, if the private signature key is used for multiple purposes outside the election (e.g. for authentication in other governmental or private services via eID), the incentive for the voter to divulge it should be much greater than for providing a receipt for her vote in one election, making large-scale vote buying less likely to occur.

Nevertheless, even in the case where the voters private signature keys are stored on trusted devices, the voting device still should be trusted not to leak the voter's choice (for vote privacy) or the fact whether the voter has cast a ballot (for participation privacy and receipt-freeness) to the adversary.

The further new assumptions that are required for our extension can be explained as follows.

(A-KTV-PosTrusteeHonest) The security of our extension relies on the adversary not being able to tell, whether a particular ballot published on the bulletin board was sent by the voter or by a posting trustee. Hence, it is crucial to trust that there is at least one posting trustee that does not disclose whether she cast a particular ballot to the adversary.

(A-KTV-AnonChannels) Similar to the previous assumption, the anonymity of the communication channels is critical in ensuring, that the adversary does not distinguish between the ballots sent by voters and the ballots sent by posting trustees. This anonymity can be ensured either by a trusted forwarding server, or onion routing.

(A-KTV-HiddenVote) If this assumption is not ensured and the adversary can observe the voter throughout all the voting phase or otherwise make sure that the voter is not able to cast her vote without adversarial observation, then it follows trivially that ensuring receipt-freeness is not possible.

(A-KTV-NoUnknownPlaintext) In our extension, this assumption relies on the voter not divulging her private signature key to the adversary, which is justified if the private signature key is also used for purposes other than voting, e.g. as a part of eID infrastructure, in which case divulging it to the adversary would incur larger losses to the voter than she would gain from selling her vote. It further relies on on the absence of two-way communication between the voter and the adversary during casting the ballot, which we consider unlikely in large-scale vote buying.

(A-KTV-NoForcedAbstention) This assumption is a weaker form of the assumption in Helios-Base, that no coercion takes place. Thus, as opposed to Helios-Base, our extension can prevent some forms of coercion (receipt-freeness), but not other ones (forced abstention and randomization).

(A-KTV-IndAbstain) This assumption, albeit a simplification from real-world behavior, is required for the sake of being able to model participation privacy.

(A-KTV-IndReceipt) As with voter independence for participation privacy, this assumption is required for the sake of being able to model receipt-freeness.

6.3 Description

In this section we describe the KTV-Helios scheme. Note that as opposed to the descriptions in Chapters 3 to 5 we provide a more formal description of the scheme in order to be able to formally evaluate its security.

Overview. The basic idea of the KTV-Helios scheme is the introduction of a new kind of entity, the posting trustee and the new type of ballots, the dummy ballots, which contain an encryption of 0. While the setup remains unchanged from Helios-Base, the voting is modified as follows. The voter encrypts the chosen voting option and submits it to the bulletin board together with the accompanying well-formedness proofs, which include the proof of plaintext knowledge as in Helios-Base, and the disjunctive proof for proving that either the voter knows the signature on the encrypted vote, or the submitted ciphertext encrypts 0. In case she later wants to update her vote for option v with the vote for another voting option v', the voter submits another ballot that encrypts $v' - v$. The ballots, composed of the encrypted vote and proof, are published on the bulletin board next to the voters identity. The posting trustee also casts a random number of dummy ballots on behalf of each voter, that are published next to that voter's identity. Each dummy ballot consists of an encryption of 0 accompanied with the well-formedness proofs that are constructed in the same way as the proofs for non-dummy ballots. The participation privacy and receipt-freeness of KTV-Helios then depends on the inability of the adversary to distinguish between the dummy ballots and the ballots cast by voters.

The tallying further differs from Helios-Base. After the voting is finished, for each voter the ballots that are published next to the voter's identity are aggregated into the final ballot. Due to the homomorphic property of the cryptosystem, and due to the fact that the dummy ballots contain the encryption of 0, this final ballot encrypts the sum of all non-dummy votes cast by the voter. The final ballots of all voters are being further anonymized via mix net shuffling. After the shuffling, each anonymized ballot is further processed by the tabulation tellers and either assigned to a valid voting option, or discarded without revealing its plaintext value.

The components of our extension (KTV-Helios) and the interactions between them are depicted on Figure 6.1.

Building Blocks of the KTV-Helios Scheme. We describe the building blocks (i.e. the cryptographic primitives, probability distributions and plaintext tally function) of the

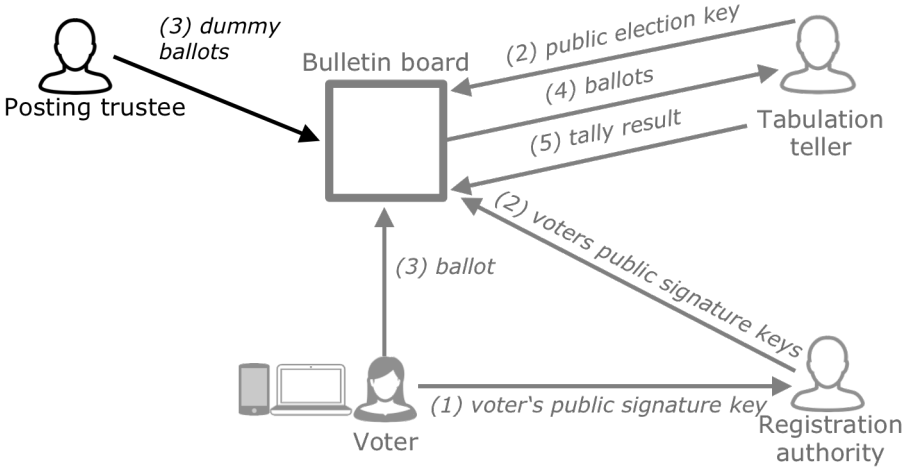


Figure 6.1: Components and their interactions of KTV-Helios. The new components compared to Helios-Base are in black. For the sake of simplicity, the verification step using the verification device is omitted, and sending the voters public signature keys to the registration authority can be skipped in case of an existing PKI.

KTV-Helios scheme. The scheme uses the following *cryptographic primitives* (see more detailed descriptions in Section 2.2):

- ElGamal with proof of plaintext knowledge [BPW12], a NM-CPA secure encryption scheme (the same one is used in Helios-Base). Its algorithms are $\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$. The encryption of a message $m \in \mathbb{Z}_q$ with a public key $(g, h) \in \mathbb{G}^2$ is $((g^r, g^m h^r), \pi_{PoK})$ where $r \leftarrow\!\!\$\, \mathbb{Z}_q$ is randomly sampled and π_{PoK} is a proof of knowledge of r (see Section 2.2.5). To decrypt a ciphertext $((c^{(1)}, c^{(2)}), \pi_{PoK})$ with a secret key sk, one first checks the validity of the proof π_{PoK} and if the proof is valid successful sets $m = c^{(2)} \cdot (c^{(1)})^{(-sk)}$.

- distributed threshold secret sharing and distributed threshold decryption for ElGamal (see Section 2.2.11). Its algorithms are $\mathsf{DistKeyGen}(N_t, t), \mathsf{DistDec}$. The algorithm $\mathsf{DistKeyGen}$ is jointly run by N_t parties in order to generate a public ElGamal key and N_t shares of a corresponding private key with a threshold of t. Correspondingly, $\mathsf{DistDec}$ run by at least t out of N_t parties in order to decrypt an ElGamal ciphertext using the previously generated private key.

- An existentially unforgeable digital signature scheme consisting of algorithms $\mathsf{SigKeyGen}$, Sign and Verify.

- The Chaum-Pedersen NIZK proof $\mathsf{EqProof}(g_1, g_2, h_1, h_2)$ that proves the equality of discrete logarithms $\log_{g_1} h_1 = \log_{g_2} h_2$ as described in [CP92] (see Sections 2.2.4 and 2.2.7). This proof can be simulated in the random oracle model, for which we write $\mathsf{SimEqProof}(g_1, g_2, h_1', h_2')$ (see e.g. [BCG+15]).

- A NIZK disjunctive proof $\mathsf{DisjProof}(\mathsf{pk}_{id}, \mathsf{sk}_{id'} \in \{\mathsf{sk}_{id}, 0\}, g_1, g_2, h_1, h_2, t)$ that given $(\mathsf{pk}_{id}, \mathsf{sk}_{id}) \leftarrow\!\!\$\ \mathsf{SigKeyGen}$ and $g_1, g_2, h_1, h_2 \in \mathbb{G}_q$ and timestamp t proves either the knowledge of $s = \mathsf{Sign}(\mathsf{sk}_s, g_1||g_2||h_1||h_2||t)$ (see Sections 2.2.4 and 2.2.8) or the equality of discrete logarithms $\log_{g_1} h_1 = \log_{g_2} h_2$.
- A verifiable re-encryption mix net for ElGamal ciphertexts $\mathsf{Mix}(c_1, ..., c_N)$ (see Section 2.2.12).
- A plaintext equivalence test (PET) for the ElGamal ciphertexts (see Section 2.2.17). On input a ciphertext c, a secret key sk and a message m, the PET outputs a decryption factor d that is 1 if c is an encryption of m under sk and random in $\mathbb{Z}_q$ otherwise. It also creates a proof π_{PET} that it operated correctly (this is another Chaum-Pedersen $\mathsf{EqProof}$, see Section 2.2.7). The PET is performed either by a single holder of a private key, or via distributed threshold decryption, if the private key is distributed among multiple parties as described in Section 2.2.11.

The next building blocks are the *probability distributions*. They are used by the posting trustees in order to cast a random number of dummy ballots at random times next to each voter's *id*. In order to specify the dummy ballot casting algorithm for the posting trustee, we use two probability distributions $\mathbb{P}_d$ and $\mathbb{P}_t$. The first probability distribution $\mathbb{P}_d$ is used to sample a number of dummy ballots for each voter. This distribution therefore has a support $[x, y]$ with x, y as the minimal and maximal number of dummy ballots that the posting trustee is going to cast for each voter (i.e., $x \in \mathbb{N}_0$, $y \in \mathbb{N} \cup \{\infty\}$). The parameters x and y, as well as the exact $\mathbb{P}_d$ needs to be chosen according to the optimal trade-off between security and efficiency in each election. The way to calculate the trade-off is provided in Section 6.4.3. The second probability distribution $\mathbb{P}_t$ is used to determine the time to cast each dummy ballot. Thus, this distribution has a support $[T_s, T_e]$ with T_s denoting the timestamp at the beginning of the voting, and T_e the timestamp at the end of the voting. In order to obfuscate the ballots cast by voters, $\mathbb{P}_t$ should be chosen so that this distribution resembles the distribution of times at which the voters cast their ballots. For this, e.g. the information from the previous elections could be used.

Next we describe the *plaintext tally function* of the KTV-Helios scheme, that takes the plaintext votes cast by voters and the posting trustee as input and outputs the election result. While this function is not actually applied in the election, its formalization is still required for proving the vote integrity of the scheme. The plaintext tally function is informally described in the following way. The valid votes cast by registered eligible voters are included in the tally. If the voter casts multiple votes, they are added together to form a final vote. If the final vote is a valid voting option, it is included in the tally, otherwise it is replaced with a null vote. If the voter abstains, their final vote is counted as a null vote[15]. The votes cast by the posting trustee are not included in the result.

[15]Note, that the function does not make a distinction between abstaining voters, and voters that cast a null vote.

The formalised description of the plaintext tally function is as follows: Let $\mathbb{G}_q$ be the plaintext space of $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$. Then, let $\mathbb{V}_{valid} = \{v_1, ..., v_L\} \subset \mathbb{G}_q^L$, $0 \notin \mathbb{V}_{valid}$ be a set of valid voting options, so that the voter is allowed to select one of these options as her vote. Let then $\rho' : (\mathbb{V}_{valid} \cup \{0\})^N \to \mathbb{N}_0^L$ be the function that, given the plaintext votes cast within the election, outputs a vector of values with the sum of cast votes for each candidate and the number of abstaining voters. Let $I = \{id_1, ..., id_N\}$ be a set of registered eligible voters, and $\hat{id} \notin I$ denote the posting trustee. Further, let N_T be the total number of votes cast within the election. We define the tally function for the KTV-Helios scheme $\rho(\mathbb{V}_{cast}) : (I \cup \{\hat{id}\} \times G_q)^* \to \mathbb{R}$ as follows:

1. Initialise a set $\mathbb{V}_{final} = \{(id_1, 0), ..., (id_N, 0)\}$
2. For every $(id, v) \in \mathbb{V}_{cast}$, if $id \in I$, replace the tuple $(id, v') \in \mathbb{V}_{final}$ with $(id, v' + v)$. If $id = \hat{id}$, discard the vote.
3. For every $(id_i, v_i) \in \mathbb{V}_{final}$, if $v_i \notin \mathbb{V}_{valid}$, replace (id_i, v_i) with $(id_i, 0)$
4. Output $\rho'(v_1, ..., v_N)$.

We further show that the function ρ provides *partial counting* property, also used for proving the vote integrity of the election analogously to [CGGI14]. The partial counting property suggests, that given a partition of the set of cast ballots, the sum of the individual tally results on each partitions should correspond to the tally result on the total set. We provide a more formal definition of partial counting as follows:

Definition 6.1. *Let $I \cup \{\hat{id}\}$ be the set of eligible voters and the posting trustee, and let the sets I_1,...,I_k partition $I \cup \{\hat{id}\}$. Let $\mathbb{V}_{cast}$ be the list of all the cast votes in the election, and define the lists $\mathbb{V}_{cast}^{(1)}, ..., \mathbb{V}_{cast}^{(k)} \subset \mathbb{V}_{cast}$ so that for each $(id, v) \in \mathbb{V}_{cast}$ holds $(id, v) \in \mathbb{V}_{cast}^{(i)} \iff id \in I_i$, $i = 1, ..., k$. A plaintext tally function f provides partial counting if it holds:*

$$f(\mathbb{V}_{cast}) = \sum_{i=1}^{k} f(\mathbb{V}_{cast}^{(i)})$$

Theorem 6.2. *The plaintext tally function of KTV-Helios ρ provides partial counting.*

Proof. Let $I = \{id_1, ..., id_N\}$ be a set of voter identities, $\hat{id} \notin I$ the identity denoting the posting trustee, $\{v_1, ..., v_L\} \in \mathbb{G}_q \setminus \{0\}$ a set of valid voting options, and let $\mathbb{V}_{cast}$ be a set of tuples (id, v) with $id \in I \cup \{\hat{id}\}$ and $v \in \mathbb{G}_q$.

Let $I_1, .., I_k$ be partitions of $I \cup \{\hat{id}\}$, so that $\bigcup_{i=1}^{k} I_i = I \cup \{\hat{id}\}$ and $I_i \cap I_j = \emptyset$ for all $i \neq j$. We further define the lists $\mathbb{V}_{cast}^{(i)} \subset \mathbb{V}_{cast}$ as a list of all the tuples $(id, v) \in \mathbb{V}_{cast}$, for which holds $id \in I_i$.

The partial counting property means, that the tally on $\mathbb{V}_{cast}$ can be expressed as a sum of tallies on all the lists $\mathbb{V}_{cast}^{(i)}$, $i = 1, ..., k$. Namely, it should hold

$$\rho(\mathbb{V}_{cast}) = \sum_{i=1}^{k} \rho(\mathbb{V}_{cast}^{(i)})$$

In order to prove this, consider the output of $\rho(\mathbb{V}_{cast}^{(i)})$. Let ρ' be the function, that, given the list of plaintext votes $v_1, ..., v_N$ outputs the number of votes for each voting option $v_1, ..., v_L$ and the number of abstaining voters. Namely, on the input of $v_1, ..., v_N \in (\{v_1, ..., v_L\} \cup 0)^{L+1} \subset \mathbb{G}_q^{L+1}$, ρ' returns a vector of values $R \in \mathbb{N}_0^{L+1}$. It holds, that ρ' supports partial counting. Namely, for two lists $S_1 = (v_{1,1}, ..., v_{N_1,1})$ and $S_2 = (v_{2,1}, ..., v_{N_2,2})$ with $S_1, S_2 \in (\{v_1, ..., v_L\} \cup 0)^{L+1}$, it holds

$$\rho'(S_1) + \rho'(S_2) = \rho'(S_1 \cup S_2)$$

According to the definition of ρ, with $\mathbb{V}$ as a set of tuples $(id, v) \in I \cup \{\hat{id}\} \times \mathbb{G}_1$, ρ outputs $R = \rho'(v_1, ..., v_N)$ with v_i, $i = 1, ..., N$ being either the sum of all votes cast by the voter $id_i \in I$, or $v_i = 0$ if there were no valid votes from the voter id_i in $\mathbb{V}$ (i.e. there is no tuple (id_i, v) with $v \in \{v_1, ..., v_L\}$ in $\mathbb{V}$).

it follows that $\rho(\mathbb{V}_{cast}^{(i)}) = \rho'(v_{1,i}, ..., v_{N,i})$ with $v_{j,i}$ denoting the sum of all cast votes by the voter id_j if $id_j \in I_i$, and $v_{j,i} = 0$ if $id_j \notin I_i$. Combined with the partial counting property of ρ' it follows that

$$\rho(\mathbb{V}_{cast}) = \sum_{i=1}^{k} \rho(\mathbb{V}_{cast}^{(i)})$$

□

Formal Description of KTV-Helios: We are now ready to provide the formal description of the KTV-Helios scheme. This description is based upon the syntax proposed in [BCG+15], adjusted to the context of the KTV-Helios scheme. For the sake of simplicity, we describe a scheme with a single posting trustee. We first specify the various functions in place, i.e.:

- $\mathsf{Register}(1^\lambda, id)$ is run by the voter and the registration authority. Given a register I of eligible voters, the function returns a pair of keys $(\mathsf{pk}_{id}, \mathsf{sk}_{id}) \leftarrow_\$ \mathsf{SigKeyGen}(1^\lambda)$ to the voter id and adds (id, pk_{id}) to the list of registered voters public signature key I_{pk} if $id \in I$.

- $\mathsf{Setup}(1^\lambda)$ is run by the tabulation tellers. If there is a single tabulation teller, the function runs $(\mathsf{pk}, \mathsf{sk}) = \mathsf{KeyGen}$ to create the election keys and returns the public election key pk. In case of $N_t > 1$ tabulation teller, the function runs $(\mathsf{pk}, \mathsf{sk}_1, ..., \mathsf{sk}_{N_t}) = \mathsf{DistKeyGen}(N_t, t)$ with $t = \lfloor N_t/2 \rfloor + 1$ and returns the public election key pk.

- $\mathsf{Vote}((id', \mathsf{sk}_{id'}), id, v, t)$ creates a ballot $b = (id, c, \pi_{PoK}, \pi, t)$ for voter $id \in I$ and voting option v, that is cast at a timestamp[16] t. If $id = id'$ (a voter casting her own ballot) then it computes $(c, \pi_{PoK}) = \mathsf{Enc}(\mathsf{pk}, v)$ where $c = (c^{(1)}, c^{(2)})$ and $\pi = \mathsf{DisjProof}(\mathsf{pk}_{id}, \mathsf{sk}_{id'}, g, h, c^{(1)}, c^{(2)}, t)$ using a signature $\mathsf{Sign}(\mathsf{sk}_{id'}, g||h||c||t)$. If $id' = \hat{id}$ (the posting trustee is casting a ballot on behalf of the voter id) then $\mathsf{sk}_{id'}$ is not required, but v must be 0. Note, that the challenges used in π_{PoK} and π should include the statements and commitments from both π_{PoK} and π in order to prevent that the voter signs and casts the ballot she did not compute herself.

- $\mathsf{Validate}(b)$ parses the ballot b as $(id, c = (c^{(1)}, c^{(2)}), \pi_{PoK}, \pi, t)$ and returns 1 if π and π_{PoK} are valid proofs, $id \in I$ and $t \in [T_s, T_e]$, and $\perp$ otherwise.

- $\mathsf{VerifyVote}(\mathsf{BB}, b)$ is used by the voter to ensure that her ballot b is properly stored on the bulletin board. It outputs 1 if $b \in \mathsf{BB}$ and $\mathsf{ValidateBB}(\mathsf{BB})$ holds, otherwise $\perp$.

- $\mathsf{VoteDummy}(id)$ is used by the posting trustees to cast dummy ballots for a given voter id. The posting trustee samples a random number $m \leftarrow_\$ \mathbb{P}_d$ and random timestamps $t_1, ..., t_m \leftarrow_\$ \mathbb{P}_t$, and returns a set of ballots

$$(\mathsf{Vote}((\hat{id}, 0), id, 0, t_1), ..., \mathsf{Vote}((\hat{id}, 0), id, 0, t_m))$$

- $\mathsf{Valid}(\mathsf{BB}, b)$ is run by the bulletin board before appending a new ballot. It checks that $\mathsf{Validate}(b) = 1$ and that the ciphertext c in b does not appear in any ballot already on the board. If this holds it returns 1, otherwise $\perp$.

- $\mathsf{ValidateBB}(\mathsf{BB})$ checks that the contents of the bulletin board are valid. It is run by the tabulation tellers as part of the tallying process and by the voters verifying the bulletin board. It creates an empty board BB' and for each ballot $b \in \mathsf{BB}$ runs "if $\mathsf{Valid}(\mathsf{BB}', b)$ then append b to BB'". If any ballot gets rejected it returns $\perp$, otherwise 1.

- $\mathsf{Tally}(\mathsf{BB}, \mathsf{sk})$ is used by the tabulation teller(s). It returns a tuple (R, Π) where R is the election result and Π is auxiliary data (proofs of correct tallying). In more detail:

  1. Run $\mathsf{ValidateBB}(\mathsf{BB})$ and return $\perp$ if this fails.
  2. Parse each ballot $b \in \mathsf{BB}'$ as $(id, c, \pi_{PoK}, \pi, t)$.
  3. For each id appearing in the ballots, set $c_{id} = \prod_{c \in C(id)} c$ where $C(id)$ is the set of ciphertexts c in ballots belonging to voter id.

[16] As the timestamp t denotes the time at which b is submitted to the bulletin board, we assume that it is chosen in $[T_s, T_e]$.

4. Mix the ballots $(c_1, \ldots, c_N)$ (where N is the number of distinct identities who cast a ballot) to get a new list of ballots $(\bar{c}_1, \ldots, \bar{c}_N)$ and a proof π_{mix} of shuffle validity. In case multiple tabulation tellers are involved, each of them performs the shuffling in their turn.
5. For each $i \in \{1, \ldots, N\}$ and each valid voting option $v \in \mathbb{V}_{valid}$, use the PET (either as a single tabulation teller or as multiple tabulation tellers using $\mathsf{DistDec}$) on $\bar{c}_i$ and v to create a decryption factor $d_{i,v}$ and proof $\pi_{PET,i,v}$.
6. The result R is the number of times each voting option was chosen, i.e. $R(v) = |\{i : d_{i,v} = 1\}|$ for all $v \in \mathbb{V}_{valid}$. The auxiliary data Π contains the proofs of shuffle validity π_{mix}, the shuffled ciphertexts $(\bar{c}_1, \ldots, \bar{c}_N)$, the decryption factors $d_{i,v}$ and the PET proofs $\pi_{PET,i,v}$ for $i \in \{1, \ldots, N\}$ and $v \in \mathbb{V}_{valid}$.

- $\mathsf{ValidateTally}(\mathsf{BB}, (R, \Pi))$ takes a bulletin board BB and the output (R, Π) of Tally and returns 1 if $\mathsf{ValidateBB}(\mathsf{BB}) = 1$ and all the proofs π_{mix} and π_{PET} are valid, otherwise $\perp$. It is used to verify an election.

These functions are combined in order to build the KTV Helios scheme. The corresponding description of the KTV Helios scheme is given in the following paragraphs along the line of the three phases of an election.

Setup. The election organizers publish a set of valid non-null voting options $\mathbb{V}_{valid} = (v_1, ..., v_L)$ with $0 \notin \mathbb{V}_{valid}$ on an empty bulletin board BB. If there is no existing PKI encompassing the eligible voters, the eligible voters from the voting register I generate and send their public signature keys to the registration authority running $\mathsf{Register}(1^\lambda, id)$, who publishes the list of registered voters $I_{pk} = \{(id_1, \mathsf{pk}_{id_1}), ..., (id_N, \mathsf{pk}_{id_N})\}$ on the bulletin board. In that case, the voters verify that the bulletin board publishes the correct public signature keys. The tabulation teller(s) run $\mathsf{Setup}(1^\lambda)$ and verify that the bulletin board publishes the correct data submitted to it.

Voting. The posting trustees run $\mathsf{VoteDummy}(id)$ for each registered eligible voter $id \in I$ independently from each other. Each posting trustee then submits each resulting dummy ballot $b = (id, c, \pi_{PoK}, \pi, t)$ to the bulletin board at a time corresponding to the timestamp t. The bulletin board appends b to BB. The voter id runs $\mathsf{Vote}((id, \mathsf{sk}_{id}), id, v, t)$ in order to cast her ballot for a voting option v at a time denoted by timestamp t. The bulletin board appends b to BB. Then, the voter can run $\mathsf{VerifyVote}(\mathsf{BB}, b)$ to verify whether her ballot is properly stored.

Tallying. The tabulation teller(s) runs $\mathsf{Tally}(\mathsf{BB}, \mathsf{sk})$ on the contents of the bulletin board, and publish the resulting output (R, Π). Everyone who wants to verify the correctness of the tally runs $\mathsf{ValidateTally}(\mathsf{BB}, (R, \Pi))$.

6.4 Security

In this section we describe the security evaluation for our extension. Namely, we propose the formal definitions of participation privacy, receipt-freeness and fairness and use existing definitions of verifiability against malicious bulletin board (to prove integrity and eligibility) and of ballot privacy (to prove vote privacy without receipt-freeness). Using these definitions, we provide formal security proofs for our scheme. In addition to this, we provide an informal evaluation for the robustness requirement.

6.4.1 Vote Privacy

In this section we prove the security of KTV-Helios for vote privacy. We do this by first evaluating KTV-Helios given the same security model for vote privacy as in Helios-Base, and then provide a proof for a stronger form of vote privacy, receipt-freeness.

Note that while our security model outlined in Section 6.2 requires more than half of the tabulation tellers to be honest, in our formal analysis we only consider the case with only one tabulation teller. Intuitively, we presume that the results should be transferable to the case with multiple tabulation tellers due to the properties of the secret sharing scheme underlying the distributed threshold secret sharing and distributed threshold decryption. Namely, possessing only less than a threshold (in our case, $t = \lfloor N_t/2 \rfloor + 1$) of the private election key shares should provide no information on the private election key or on the plaintexts encrypted with a corresponding public election key. Similar to [BCG+15], the security proofs for such a case will be considered in future work.

Vote Privacy without Receipt-Freeness

We first prove vote privacy in our security model for the KTV-Helios scheme for the voters who do not attempt to create a receipt for their vote. For this, we use the definition of ballot privacy (BPRIV) in [BCG+15]. Since the original definition also uses two auxiliary properties called strong correctness and strong consistency, we prove these as well. Together these definitions imply that an adversary does not get more information from an election scheme as they would from the election result alone. Put differently, the election data — ballots on the bulletin board, well-formedness proofs, proofs of shuffle validity and of decryption validity — do not leak any information about the votes. We assume like in [BCG+15] that both the tabulation teller and the bulletin board that the voter communicates with are honest (assumptions (A-KTV-TabTellerHonest), (A-KTV-NoBBModification), (A-KTV-BBConsistency) and (A-KTV-Verify)), the voting device does not leak private information (A-KTV-VotDeviceLeakage) and the adversary is computationally restricted (A-KTV-CompRestricted), which corresponds to the definition of vote privacy (without receipt-freeness) and the security assumptions we require for its fulfillment as outlined in Section 6.2.

Purpose and Definition of BPRIV: We adjust the definition proposed by Bernhard et al. [BCG+15] – more precisely the definition in the random oracle model – to the KTV-Helios scheme by including additional parameters required for casting a ballot. We also omit the Publish algorithm as our bulletin boards do not store any non-public data (our Publish would be the identity function). Recall that a scheme satisfies BPRIV [BCG+15] if there exists an algorithm SimProof such that no adversary has more than a negligible chance of winning the BPRIV game; the game itself uses the SimProof algorithm in the tallying oracle.

The purpose of BPRIV is to show that one does not learn anything more from the election data (including the bulletin bulletin board and any proofs output by the tallying process) than from the election result alone. In other words, the election data does not leak information about the votes, at least in a computational sense[17]. For example, if Alice, Bob and Charlie vote in an election and the result is "3 yes" then the result alone implies that Alice must have voted yes, which is not considered a privacy breach. But if Charlie votes yes and the result is "2 yes, 1 no" then Charlie should not, without any further information, be able to tell whether Alice voted yes or no as this does not follow from the result.

The BPRIV notion is a security experiment with two bulletin boards, one of which (chosen at random by sampling a bit β) is shown to the adversary. For each voter, the adversary may either cast a ballot themselves or ask the voter to cast one of two votes v_0, v_1 in which case a ballot for v_0 is sent to the first bulletin board and a ballot for v_1 is sent to the second bulletin board. The adversary thus sees either a ballot for v_0 or a ballot for v_1 and a scheme is BPRIV secure if no PPT adversary has better than a negligible chance of distinguishing the two cases. At the end of the election, the adversary is always given the election result for the first bulletin board. This disallows trivial wins if the adversary makes the results on the two bulletin boards differ from each other. If the first bulletin board was the one shown to the adversary, it is tallied normally; if the adversary saw the second bulletin board but the first result then the experiment creates fake tallying proofs to pretend that the second bulletin board had the same result as the first one. This is the role of the SimProof algorithm that must be provided as part of a BPRIV security proof.

The experiment $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ for the scheme $\mathcal{S}$ is formally defined as follows: The challenger sets up two empty bulletin boards BB_0 and BB_1, runs the setup phase as outlined in Section 6.3 and publishes the public election key pk. The challenger also chooses a random $\beta \in \{0, 1\}$. The adversary can read the bulletin board BB_β at any time and can perfomr the following oracle queries:

- $\mathcal{O}\mathsf{Cast}(b)$: This query lets the adversary cast an arbitrary ballot b, as long as b is

[17] In an information-theoretic sense, a ballot with an encrypted vote does of course contain information about the vote, otherwise one could not tally it. But since the votes are encrypted, they should not help anyone who does not have the private election key to discover the contained vote.

valid for the bulletin board BB_β that the adversary can see. If $\mathsf{Valid}(\mathsf{BB}_\beta, b) = 1$, the challenger runs $\mathsf{Append}(\mathsf{BB}_0, b)$ and $\mathsf{Append}(\mathsf{BB}_1, b)$ to append the ballot b to both bulletin boards.

- $\mathcal{O}\mathsf{VoteLR}(id', id, v_0, v_1, t)$: This lets the adversary ask a voter to vote for either v_0 or v_1 depending on the secret β. First, if $id \in I$ and $id' = id$ the challenger computes $b_0 = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, v_0, t)$ and $b_1 = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, v_1, t)$. If $id \in I$ and $id' = \hat{id}$ then the challenger computes two[18] ballots $b_0 = \mathsf{Vote}((id', \mathsf{sk}_{id'}), id, 0, t)$ and $b_1 = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, 0, t)$. If none of these cases applies, the challenger returns $\perp$.
  Secondly, the challenger checks if $\mathsf{Valid}(\mathsf{BB}_\beta, b_\beta) = 1$ and returns $\perp$ if not. Finally the challenger runs $\mathsf{Append}(\mathsf{BB}_0, b_0)$ and $\mathsf{Append}(\mathsf{BB}_1, b_1)$.

- $\mathcal{O}\mathsf{Tally}()$: The adversary calls this to end the voting and obtain the results. They may call this oracle only once and after calling it, the adversary may not make any more $\mathcal{O}\mathsf{Cast}$ or $\mathcal{O}\mathsf{VoteLR}$ calls.
  The challenger computes a result and auxiliary data for BB_0 as $(R, \Pi) = \mathsf{Tally}(\mathsf{BB}_0, \mathsf{sk})$. If $\beta = 1$, the challenger also computes simulated auxiliary data for BB_1 as $\Pi = \mathsf{SimProof}(\mathsf{BB}_1, R)$, overwriting the previous auxiliary data Π. The challenger then returns (R, Π) to the adversary.

At the end, the adversary has to output a guess $g \in \{0, 1\}$. We say that the adversary wins an execution of the experiment if $g = \beta$.

Definition 6.3. *A voting scheme $\mathcal{S}$ satisfies ballot privacy (BPRIV) if there exists a PPT simulation function $\mathsf{SimProof}(\mathsf{BB}, R)$ so that for any PPT adversary the quantity*

$$\mathsf{Adv}^{\mathsf{bpriv}}_{\mathcal{A},\mathcal{S}} := \left| \Pr\left[\mathsf{Exp}^{\mathsf{bpriv},0}_{\mathcal{A},\mathcal{S}} = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{bpriv},1}_{\mathcal{A},\mathcal{S}} = 1\right] \right|$$

is negligible (in the security parameter).

Proof for the KTV-Helios Scheme: The core of a BPRIV proof is a simulator $\mathsf{SimTally}$ that, when $\beta = 1$, takes as input the bulletin board BB_1 and the result R from BB_0 and outputs simulated data Π that the adversary cannot distinguish from real auxiliary data, such as proofs of shuffle validity or of decryption validity. This proves that the auxiliary data Π does not leak any information about the votes, except what already follows from the result.

Recall that the tallying process in KTV-Helios is as follows:

1. Remove any invalid ballots from the bulletin board using $\mathsf{ValidateBB}$.

[18] Vote is a randomised algorithm so the effect of calling it twice on the same inputs is to create two distinct ballots.

2. Homomorphically aggregate the ballots from each voter.
3. Shuffle the remaining ballots (one per voter) in a mix-net.
4. Match each shuffled ballot against each valid vote $v \in V$ with a PET.
5. Compute the number of voters who chose each vote $v \in V$ by counting the successful PETs. This gives the election result R.
6. The auxiliary data Π comprises the proofs of shuffle valiidity Π_{mix} from stage 3 and the data and proofs Π_{PET} forming the PETs in stage 4.

The additional PET stage compared to (non-KTV) Helios actually makes the ballot privacy proof easier. The simulator $\mathsf{SimProof}(\mathsf{BB}, R)$ works as follows:

1. Remove any invalid ballots from the bulletin board BB using $\mathsf{ValidateBB}$.
2. Homomorphically aggregate the ballots from each voter.
3. Shuffle the remaining ballots (one per voter) in a mix-net. Note, we do not need to simulate the mix-net; we can just run a normal mix (and store the auxiliary data Π_{mix} that this creates).
4. Simulate the PETs (we will describe this in detail below) to get simulated data Π'_{PET}.
5. Return (Π_{mix}, Π'_{PET}).

The following lemma is useful to construct the PET simulator.

Lemma 6.4. *In any execution of the BPRIV game, if we tallied both bulletin boards then with all but negligible probability, both bulletin boards would end up with the same number of ballots.*

Note that both the $\mathcal{O}\mathsf{VoteLR}$ and the $\mathcal{O}\mathsf{Cast}$ oracles either add one ballot to both bulletin boards each or do not add any ballots at all. Therefore we have the invariant that the number of ballots before tallying is the same on both bulletin boards with probability 1.

The first stage of the tallying algorithm runs $\mathsf{ValidateBB}$ to remove possibly invalid ballots. On the visible bulletin board BB_β, since all ballots were already checked in the oracles before placing them on the bulletin board, we conclude that $\mathsf{ValidateBB}$ does not remove any ballots. On the invisible bulletin board $\mathsf{BB}_{(1-\beta)}$, if any ballot b gets removed then we consider the query (VoteLR or Cast) where it was created. The only way a ballot b can get removed again is if at the time it was added, it was valid on BB_β (or it would never have got added at all) but invalid on $\mathsf{BB}_{(1-\beta)}$ (or it would not get removed again later). But this means that the ciphertext c in the ballot b in question must be a copy of an earlier ciphertext on $\mathsf{BB}_{(1-\beta)}$ but not on BB_β, as this is the only other case when Valid declares

a ballot invalid, and the only such ballots are those created by $\mathcal{O}\mathsf{VoteLR}$. Therefore we conclude that either two ballots created by $\mathcal{O}\mathsf{VoteLR}$ have collided, the probability of which is certainly negligible, or the adversary has submitted in a $\mathcal{O}\mathsf{Cast}$ query a copy of a ciphertext that $\mathcal{O}\mathsf{VoteLR}$ previously placed on the invisible bulletin board $\mathsf{BB}_{(1-\beta)}$. Since the adversary never saw this ciphertext, and since the encryption scheme is NM-CPA so ciphertexts must be unpredictable, the probability of this event is negligible too. This concludes the proof of Lemma 6.4.

We now describe how to simulate the PET. Our inputs are a number N of ballots (the output of the mix-net), a result R that was correctly computed on a different bulletin board that also had N ballots (after stage 1 of tallying) by Lemma 6.4 and a set $\mathbb{V}_{valid}$ of valid voting options.

Since the PETs in a real tally are taken over ballots that have just come out of a mix-net, the distribution of votes in these ballots is a uniformly random permutation of votes subject to the tally being R. For example, if R indicates that there was one vote for $v = 1$ and $N - 1$ votes for $v = 2$ then the probability of the 1-vote being in the i-th ballot is $1/N$, irrespective of the order in which the ballots were cast (for example the adversary might know that the first person to vote was the one that cast the 1-vote). This is because the ballots are uniformly permuted in the mix-net.

Our simulation strategy is therefore to emulate this random permutation. The result R gives us a mapping $f_R : \mathbb{V}_{valid} \cup 0 \to \{0, 1, \ldots, n\}$ where for example $f_R(v) = 3$ means that three voters voted for v and $f_R(0)$ is the number of voters who cast an invalid vote or abstained. We have $f_R(0) + \sum_{v \in \mathbb{V}_{valid}} f_R(v) = N$, i.e. the number of invalid/abstention votes plus the totals for each valid option sum to the number n of ballots that came out of the mix-net. We simulate as follows:

1. Create a list $L = (L_1, \ldots, L_N)$ such that each vote $v \in \mathbb{V}_{valid}$ appears $f_R(v)$ times in L and the value 0 appears $f_R(0)$ times. Then permute L randomly.

2. Create an $N \times |V|$ matrix d of PET results: if $L[i] = v$, which means that we pretend voter i voted for $v \in \mathbb{V}_{valid}$, then set $d_{i,v} = 1$. Otherwise set $d_{i,v}$ to be a random element of $\mathbb{Z}_q$.

3. For each (i, v) pair create a simulated PET proof as follows. For each ciphertext $c_i = (c_i^{(1)}, c_i^{(2)})$ and each valid voting option $v \in \mathbb{V}_{valid}$ pick a random $r_{i,v} \leftarrow\!\!\$\, \mathbb{Z}_q$ and set $s_{i,v} = ((c_i^{(1)})^r, (c_i^{(2)}/v)^r)$. Then compute proofs

$$\pi_{i,v} = \mathsf{SimEqProof}(g, s_{i,v}^{(1)}, h, s_{i,v}^{(2)}/d_{i,v}) \cup \mathsf{EqProof}(c_i^{(1)}, c_i^{(2)}/v, s_{i,v}^{(1)}, s_{i,v}^{(2)})$$

4. Return the mix-net proofs of shuffle validity Π_{mix} and the PET proofs/data Π_{PET} consisting of the values $d_{i,v}$, $s_{i,v}$ and the associated proofs $\pi_{i,v}$.

The $\mathsf{EqProof}$ part proves that the $s_{i,v}$ are correct re-randomisations of the c_i for the votes $v \in \mathbb{V}_{valid}$, which they are. The $\mathsf{SimEqProof}$ are fake proofs that the $d_{i,v}$ are the

decryptions of the $s_{i,v}$ which is generally false since we chose the $d_{i,v}$ values randomly. As the encryption scheme in question is NM-CPA secure, no PPT adversary has more than a negligible change of telling a correct d-value from a false one without any proofs (indeed, this is why we have the proofs of decryption validity in the real tally) and since the proofs are zero-knowledge, we can assume that a PPT adversary cannot tell a real from a simulated proof. Therefore the proofs $\pi_{i,v}$ do not help in distinguishing real from fake $d_{i,v}$ either.

The adversary does know the result R (since the challenger in the BPRIV game outputs that and $\mathsf{SimTally}$ cannot change it) but the simulated decryptions $d_{i,v}$ are consistent with R and follow the same distribution as the real ones. Therefore we can claim that the output of the tallying oracle in case $\beta = 1$ is indistinguishable to PPT adversaries from the output in the case $\beta = 0$. The other information that the adversary sees are the ballots on the bulletin board (in particular the $\mathcal{O}\mathsf{VoteLR}$ ones which have a dependency on β) but these are ciphertexts in an NM-CPA secure encryption scheme so we can assume that they are indistinguishable to PPT adversaries too. We therefore conclude that KTV-Helios satisfies BPRIV and have proven the following.

Theorem 6.5. *KTV-Helios satisfies the BPRIV security definition.*

Strong Correctness and Strong Consistency: Together with BPRIV, [BCG+15] contains two auxiliary properties called strong correctness and strong consistency that are also required for a voting scheme to guarantee vote privacy. We define and check these properties here for the KTV scheme.

The Valid algorithm can reject new ballots based on the information already on the bulletin board (for example, it can reject a duplicate of an existing ballot). Strong correctness ensures that the rejection algorithm is not too stong, in particular that dishonest voters cannot manipulate the bulletin board to the point where it would prevent an honest voter from casting her ballot. To model this we let the adversary choose a bulletin board and test if an honest ballot, for which the adversary can choose all inputs, would get rejected from this bulletin board.

Since the original definition did not contain timestamps or a list of registered voter identities, we adapt the syntax of the original definition [BCG+15, Def. 9] to include these elements.

Definition 6.6. *A voting scheme S has strong correctness if no PPT adversary has more than a negligible probability of winning in the following experiment.*

1. *The challenger sets up the voting scheme and publishes the public election key* pk *and the list of voter identities and public keys I.*

2. *The adversary generates a bulletin board* BB*, a voter identity $id \in I$, a vote $v \in V$ and a timestamp $t \in [T_s, T_e]$.*

3. *The challenger creates a ballot* $b = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, v, t)$.

4. *The adversary loses if there is already a ballot with timestamp* $t' \geq t$ *on* BB.

5. *The adversary wins if* $\mathsf{Valid}(\mathsf{BB}, b)$ *rejects the honest ballot* b.

We have made the following changes compared to the original definition: we have added identities id to match the syntax of our voting scheme and demanded that the adversary choose an $id \in I$ since otherwise the ballot b will quite legitimately be rejected. We have also added timestamps and the restriction that the adversary must choose a timestamp t satisfying both $t \leq T_e$ and $t > t'$ for any timestamp t' of a ballot already on the bulletin board BB. Otherwise one could trivially stop any more ballots from being accepted by putting a ballot with timestamp T_e on the bulletin board.

Lemma 6.7. *The voting scheme described in Section 6.3 satisfies strong correctness.*

Proof. If $\mathsf{Valid}(\mathsf{BB}, b)$ fails on a ballot then one of two things must have happened: $\mathsf{Validate}(b) = 0$ or the ciphertext c in b is already on the bulletin board somewhere.

$\mathsf{Validate}(b)$ only fails if the identity id in b is not in I, one of the proofs in b does not verify or the timestamp is out of its domain. Since we are considering a honestly generated ballot b in the strong correctness experiment, correctness of the proof schemes involved means that the proofs are correct.

Since the ballot b in question is created by Vote which picks a fresh random $r \leftarrow\!\!\$\, \mathbb{Z}_q$, the probability of c colliding with a previous ciphertext (even an adversarially created one) is negligible. (To be precise, since we are assuming a PPT adversary, the bulletin board BB created by the adversary can only contain a polynomially bounded number of ciphertexts and since the probability of a collision with any of these is negligible individually, so is the sum of these probabilities for a union bound.) This proves Lemma 6.7. □

The definition of strong correctness may seem tautological (and the proof trivial) but it prevents the following counter-example from [BCG+15, Section 4.4]: an adversary can set a particular bit in a ballot of its own that causes the bulletin board to reject all further ballots. Assuming that either Alice wants to vote for (candidate) 1 and Bob wants to vote for 2 or the other way round, in a private voting scheme we would not expect the adversary to be able to tell who voted for 1. Without strong correctness, the adversary could let Alice vote then submit their "special" ballot to block the bulletin board, then ask Bob to vote. Since Bob's ballot now gets rejected, the result is exactly Alice's vote, so the adversary discovers how she voted.

Strong consistency prevents the Valid algorithm from leaking information in scenarios such as the following: the adversary can submit a special ballot that gets accepted if and only if the first ballot already on the bulletin board is a vote for 1. Of course this is mainly of interest where Valid has access to non-public information, either because it has access to a private election key or the bulletin board contains non-public information.

Strong consistency formally says that the election result is a function of the votes and that each valid ballot must be uniquely associated with a vote. In particular, the vote in one ballot cannot depend on the other ballots on the bulletin board.

Definition 6.8. *A voting scheme has strong consistency relative to a result function ρ if there are two algorithms*

- $\mathsf{Extract}(\mathsf{sk}, b)$ *takes an private election key and a ballot and returns either a pair (id, v) containing an identity $id \in I$ and a vote $v \in V$, or the symbol $\perp$ to denote an invalid ballot.*
- $\mathsf{ValidInd}(\mathsf{pk}, b)$ *takes an public election key and a ballot and returns* 0 *(invalid ballot) or* 1 *(valid ballot).*

such that the following conditions hold.

1. *The extraction algorithm returns the identity and vote for honestly created ballots: for any election key $(\mathsf{pk}, \mathsf{sk})$ created by Setup, any voter registration list I and any ballot b created by $\mathsf{Vote}((id, \mathsf{sk}_{id}), id, v, t)$ where $id \in I$, $t \in [T_s, T_e]$ and $v \in V$ we have $\mathsf{Extract}(\mathsf{sk}, b) = (id, v)$.*
2. *Ballots accepted onto a bulletin board are also accepted by $\mathsf{ValidInd}$: for any bulletin board BB, if $\mathsf{Valid}(\mathsf{BB}, b)$ holds then $\mathsf{ValidInd}(\mathsf{pk}, b)$ holds too.*
3. *For any PPT adversary A, the probability of winning the following game is negligible:*
   a) *Create an election key $(\mathsf{pk}, \mathsf{sk})$ with Setup and set up user registration list I.*
   b) *Give A the all public* and private *election key, as well as public and private signature keys of the voters and let A return a bulletin board BB. Let n be the number of ballots on this bulletin board.*
   c) *A loses if there is any ballot b on the bulletin board BB for which $\mathsf{ValidInd}(b) = 0$.*
   d) *Let $(r_1, \Pi) = \mathsf{Tally}(\mathsf{sk}, \mathsf{BB})$. The adversary loses if the tallying function returns $\perp$.*
   e) *Let $e_i = \mathsf{Extract}(b_i)$ for $i = 1, \ldots, n$ and the b_i are the ballots on the bulletin board BB. Let $r_2 = \rho(e_1, \ldots, e_n)$.*
   f) *The adversary wins if $r_1 \neq r_2$.*

We prove that KTV-Helios satisfies strong consistency. This means that we have to check that the tally function really counts the votes in the ballots.

For $\mathsf{Extract}(\mathsf{sk}, b)$ we parse the ballot as $b = (id, c, \pi_{PoK}, \pi, t)$ and check the two proofs; if either of them fail then we return $\perp$. Then we decrypt c with sk to get a vote v. If $v \in \mathbb{V}_{valid}$ then we return (id, v) otherwise we return $\perp$. For $\mathsf{ValidInd}(\mathsf{pk}, b)$ we just run $\mathsf{Validate}(b)$. We can assume that the list I of voter identities and public signature keys is public. We check the three conditions:

1. This follows from correctness of the encryption and proof schemes. If we encrypt a vote $v \in V$ to get ciphertext c then we also get v back when we decrypt c with the matching key and the correct voting algorithm produces correct proofs too.

2. Since Valid runs Validate, it must hold that ballots accepted onto the bulletin board are valid.

3. In fact the probability of an adversary winning this game is zero. Consider an execution of the experiment in which $r_1 \neq r_2$ in the last stage. We know that Tally did not return $\perp$ or we would not have got this far, therefore all ballots on the bulletin board passed Validate individually and the bulletin board as a whole passed ValidBB(BB). In particular ValidateBB did not cause tallying to abort.

   In this case, by the definition of Tally, the result r_1 is obtained by homomorphically adding the ciphertexts of each voter, mixing (which does not change the votes) and then PET-decrypting the resulting ballots which for all valid votes produces the same result as normal decryption whereas invalid ones are discarded.

   The extraction to get r_2 on the other hand first decrypts each ciphertext individually, then (to evaluate ρ) sums the decrypts for each voter, discards invalid sums and then reports the number of votes for each option. By the homomorphic property of the encryption scheme, these two methods of tallying must return the same result r (strong consistency does not deal with the proofs Π of correct tallying).

This concludes the proof of strong consistency. □

Receipt-Freeness

We now prove that KTV-Helios ensures receipt-freeness as a stronger form of vote privacy for the voters who might try to create a receipt for their vote. The KTV-Helios scheme ensures probabilistic receipt-freeness via deniable vote updating. The principle of deniable vote updating has also been proposed in other e-voting schemes [LH16,LHK16,AKLMQ15] in order to protect against vote selling and to prevent a voter from constructing receipts that show how the voter has voted. As such, the voter can cast her ballot for the voting option the adversary instructs to vote for, but due to deniable vote updating the voter can change her vote without the adversary knowing it.

The variant of deniable vote updating used in KTV-Helios is also characterized by enabling the so-called preliminary deniable vote updating. Given two ballots $b_{\mathcal{A}}$, b_v, with $b_{\mathcal{A}}$ as the ballot with the vote for a candidate demanded by the adversary, and b_v the ballot that "updates" $b_{\mathcal{A}}$ to a vote for a candidate chosen by the voter, the voter can cast $b_{\mathcal{A}}$ and b_v in any order. This approach prevents an attack, where the voter succeeds to cast $b_{\mathcal{A}}$ as the last ballot in the election, thus making sure that her vote has not been updated. However, in KTV-Helios, constructing b_v requires the knowledge of a vote that was cast with $b_{\mathcal{A}}$.

We propose a formal definition for probabilistic receipt-freeness for e-voting schemes with deniable vote updating. Our definition is inspired by the definition of coercion resistance by Kuesters et al. in [KTV10a] and the definition of receipt-freeness by Cortier et al. [CFG15, CCFG16]. As such, we introduce a game-based definition based on [CFG15] and modified for the support of deniable vote updating. Similar to [KTV10a], we employ the δ-notation in order to denote an adversarial advantage δ in finding out whether the voter indeed voted as instructed by the adversary, or whether she faked the receipt and voted for another voting option. Furthermore, similar to [KTV10a], we consider vote buying from a single voter, while considering an extension towards multiple voters in future work.

Note that the definition in [CFG15,CCFG16] argues that the receipt-freeness should not rely on the actions of the voter, the so-called "counter-strategy", that the voter should apply in order to fake her receipt while still voting how she wants to. Indeed, as outlined in the overview of the related work in Section 6.5, different approaches exist on whether receipt-freeness should include counter-strategies or not. Hence, we agree that our definition does not encompass this type of strong receipt-freeness, but describes a weaker version of it instead, which is ensured in KTV-Helios and other schemes that rely on deniable vote updating.

Thus, we adjust the definition by Cortier et al. by enabling the voter to apply a counter-strategy against an adversary that demands a receipt, namely, to deniably update her vote. The receipt-freeness in our definition relies on the existence of following algorithms:

- $\mathsf{DeniablyUpdate}(id, \mathsf{sk}_{id}, v_0, v_1, t_v)$ as the function for casting a ballot that changes the vote of the voter id from v_0 to v_1. The function further takes as input the voter's private signature key sk_{id} and the timestamp at which the updating ballot is cast. For KTV-Helios, $\mathsf{DeniablyUpdate}(id, \mathsf{sk}_{id}, v_0, v_1, t_v)$ is defined as casting a ballot for $v_1 - v_0$, that is

$$\mathsf{DeniablyUpdate}(id, \mathsf{sk}_{id}, v_0, v_1, t_v) = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, v_1 - v_0, t_v)$$

- $\mathsf{Obfuscate}(id)$ as the function used by the voting system for hiding the presence of ballots cast by the voter id for the purpose of deniable vote updating. For KTV-Helios, $\mathsf{Obfuscate}(id)$ models the output of an honest posting trustee and is defined as casting a random number of dummy ballots distributed according to $\mathbb{P}_d$, $\mathbb{P}_t$, that is

$$\mathsf{Obfuscate}(id) = \mathsf{VoteDummy}(id)$$

- $\mathsf{SimProof}(\mathsf{BB}, R)$ as the function for simulating the proof of correct tallying given the ballots published on the bulletin board BB and the tally result R. For KTV-Helios, $\mathsf{SimProof}(\mathsf{BB}, R)$ is defined in the same way as described in the evaluation of ballot privacy.

We define an experiment $\mathsf{Exp}^{\mathsf{rfree},\beta}_{\mathcal{A},\mathcal{S}}$ for a voting scheme $\mathcal{S}$ as follows:

The challenger sets up two bulletin boards BB_0, BB_1 by running the setup as described Section 6.3 and randomly chooses $\beta \in \{0,1\}$, so that the adversary only sees BB_β. The adversary has access to the following queries:

- $\mathcal{O}\mathsf{VoteLR}(id, v_0, v_1, t)$: the oracle computes $b_0 = \mathsf{Vote}(id, \mathsf{sk}_{id}, v_0, t)$ and $b_1 = \mathsf{Vote}(id, \mathsf{sk}_{id}, v_1, t)$ and appends b_0 to BB_0, b_1 to BB_1.
- $\mathcal{O}\mathsf{Receipt}(id, v_0, v_1, t)$: the oracle returns $\perp$ if $v_0 \notin \mathbb{V}_{valid}$, that is, if an adversary wants to obtain a receipt for casting a vote for a non-valid voting option. Otherwise, the oracle computes $b_{\mathcal{A}} = \mathsf{Vote}(id, \mathsf{sk}_{id}, v_0, t)$ and appends $b_{\mathcal{A}}$ to both of the bulletin boards BB_0, BB_1. The oracle furthermore returns the random coins ω used for constructing $b_{\mathcal{A}}$ to the adversary. Additionally, the oracle computes $b_v = \mathsf{DeniablyUpdate}(id, \mathsf{sk}_{id}, v_0, v_1, t_v)$ for a random timestamp $t \leftarrow\!\!\$\, \mathbb{P}_t$ and appends b_v to BB_1. The oracle further runs an obfuscation algorithm $\mathsf{Obfuscate}(id)$ on both of the bulletin boards. The adversary is allowed to query $\mathcal{O}\mathsf{Receipt}$ only once.
- $\mathcal{O}\mathsf{Tally}$: The oracle returns the tally result R on BB_0 and the auxiliary data Π which is either real in case $\beta = 0$ or simulated (i.e. $\Pi = \mathsf{SimProof}(\mathsf{BB}_1, R)$) in case $\beta = 1$. The adversary is allowed to query $\mathcal{O}\mathsf{Tally}$ only once.

The oracle further fills both of the bulletin boards with the content on behalf of honest voters and honest voting system entities. At the end of an experiment, the adversary has to output her guess for β.

Intuitively, the definition encompasses the scenario of vote selling, whereby the adversary tells the voter the name of the candidate the voter has to provide a receipt for, and the voter is able to access the random coins used in creating an adversarial ballot $b_{\mathcal{A}}$ (i.e. the randomness used in creating an ElGamal ciphertext). It, however, does not cover the scenarios where the adversary wants to make sure the voter did not cast a valid vote in the election, or to change the voter's vote to a random candidate (forced abstention and randomization as described in [JCJ05]). It also does not consider the information leakage from the election result.

We are now ready to define δ-receipt-freeness for deniable vote updating:

Definition 6.9. *The voting scheme $\mathcal{S}$ achieves delta-receipt-freeness, if there are algorithms* SimProof, DeniablyUpdate, Obfuscate *so that holds*

$$|\Pr\left[\mathsf{Exp}^{\mathsf{rfree},0}_{\mathcal{A},\mathcal{S}} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{rfree},1}_{\mathcal{A},\mathcal{S}} = 0\right] - \delta|$$

is negligible in the security parameter.

The assumptions regarding adversarial capabilities for receipt-freeness in KTV-Helios are then as follows: the tabulation teller does not divulge her private election key to the adversary (A-KTV-TabTellerHonest), both the voting and the verification device do not leak

the information to the adversary (A-KTV-VotDeviceLeakage, A-KTV-VerDeviceTrusted), the adversary is incapable of observing the communication channel between the voter, the posting trustees and the voting system (A-KTV-AnonChannels), at least one posting trustee does not divulge private information to the adversary (A-KTV-PosTrusteeHonest), the voter verifies that she communicates with an authentic bulletin board during voting, the bulletin board does not remove or modify the data published on it and shows the same contents to everyone (A-KTV-NoBBModification, A-KTV-BBConsistency), the voter is capable of casting a vote without being observed by the adversary (A-KTV-HiddenVote) and the voters who are required by the adversary to provide receipts act independent from each other (A-KTV-IndReceipt). Hence, the assumptions match the ones given in Section 6.2.

In order to find an appropriate value of δ, so that we can show that KTV-Helios achieves δ-receipt-freeness, we further need to account for the adversarial advantage gained from the number of ballots next to voter's identity on the bulletin board. For this purpose, we define the following experiment $\mathsf{Exp}^{\mathsf{rfnum},\beta}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t}$: The challenger chooses a random $\beta\{0,1\}$ and outputs the number $m+\beta$, with $m \leftarrow\!\!\$\, \mathbb{P}_d$, and the set of timestamps $t_1, ..., t_m, t_{m+\beta}$ that are independently sampled from $\mathbb{P}_t$ to the adversary. The adversary has to guess β. Hence, the experiment models the voter either obeying the adversary's instructions (for $\beta = 0$) or casting an additional ballot (for $\beta = 1$), whereby the adversary only has access to the number of ballots and their timestamps, but not to the ballots themselves.

Let $\delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$ denote an advantage in this experiment, so that

$$\Pr\left[\mathsf{Exp}^{\mathsf{rfnum},0}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{rfnum},1}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t} = 0\right] - \delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$$

is negligible. We are now ready to provide an evaluation of δ-receipt-freeness for KTV-Helios.

Theorem 6.10. *KTV-Helios, instantiated with probability distributions $\mathbb{P}_d, \mathbb{P}_t$, achieves δ-receipt-freeness privacy given the algorithms* SimProof, DeniablyUpdate, Obfuscate, *with $\delta = \delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$. It further does not achieve δ'-receipt-freeness for any $\delta' < \delta$.*

Proof. We base our proof on the idea, that the number of ballots next to the voter is the only source of information that gives advantage to the adversary. We consider a sequence of games, starting from $\mathsf{Exp}^{\mathsf{rfree},0}_{\mathcal{A}}$ and ending with $\mathsf{Exp}^{\mathsf{rfree},1}_{\mathcal{A}}$ and show, that the adversary $\mathcal{A}$ distinguishes the transition through all those games with the advantage of at most $\delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$. We define $\mathsf{BB}_{0,i}$ as the content of the bulletin board and (R_i, Π_i) as the tally output at the end of the game G_i, $i = 1, ..., 4$. We define the sequence as follows:

- G_1**.** The first game G_1 is equivalent to the experiment $\mathsf{Exp}^{\mathsf{rfree},\beta}_{\mathcal{A}}$ with $\beta = 0$ (hence, it is equivalent to the election where the voter *id* does not try to deniably update her vote). Thus, the content of $\mathsf{BB}_{0,1}$ and the tally output (R_1, Π_1) correspond to the content of BB_0 and the output of $\mathcal{O}\mathsf{Tally}$ at the end of $\mathsf{Exp}^{\mathsf{rfree},0}_{\mathcal{A}}$.

- G_2. The second game G_2 is equivalent to the election, where the voter id casts an additional ballot with a null-vote. Thus, the content of the bulletin board $\mathsf{BB}_{0,2}$ is equivalent to the content of the bulletin board BB_1 at the end of $\mathsf{Exp}^{\mathsf{rfree},1}_{\mathcal{A}}$ for the adversary using the query $\mathcal{O}\mathsf{Receipt}(id, v_0, v_1, t)$ with $v_0 = v_1$.

We prove, that the adversary has an advantage of δ_{num} of distinguishing between the output of G_1 and G_2. The tally result does not change, hence the tally output (R_2, Π_2) is equivalent to the tally output (R_1, Π_1). The only difference between the contents of $\mathsf{BB}_{0,1}$ and $\mathsf{BB}_{0,2}$ are the ballots next to id. Namely, G_1 contains only the ballot $b_{\mathcal{A}}$ and m dummy ballots $b_1, ..., b_m$ generated by the function $\mathsf{VoteDummy}(id)$ next to id, with $m \leftarrow_{\$} \mathbb{P}_d$ and the timestamps for the ballots $b_1, ..., b_m$ randomly sampled from $\mathbb{P}_t$. As for the second game, in addition to the ballots $b_{\mathcal{A}}$, $b_1, ..., b_m$, the bulletin board $\mathsf{BB}_{0,2}$ further contains an additional non-dummy (i.e. cast by the voter, not by a posting trustee) ballot $b_v = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, 0, t_v)$ cast by the voter at a random timestamp $t_v \leftarrow_{\$} \mathbb{P}_t$. As b_v, as well as $b_1, ..., b_m$, contains an encryption of 0, and due to the zero-knowledge property of the disjunctive proof π attached to both dummy and non-dummy ballots, it holds that b_v is indistinguishable from the dummy ballots $b_i, ..., b_m$. Furthermore, the timestamp attached to b_v is randomly sampled from the same distribution $\mathbb{P}_t$ as the timestamps for the dummy ballots $b_1, ..., b_m$. Hence, the number of the ballots next to id remains the only source of information that the adversary can use to gain advantage in distinguishing between G_1 and G_2.

It therefore follows, that in order to distinguish between G_1 and G_2, the adversary has to distinguish, given the number of ballots m', whether m' was sampled from $\mathbb{P}_d$ (in which case the adversary is in G_1), or $m' = m + 1$ with $m \leftarrow_{\$} \mathbb{P}_d$ (in which case there is an additional non-dummy ballot, and the adversary is in G_2). This distinction corresponds to the definition of the experiment $\mathsf{Exp}^{\mathsf{rfnum},\beta}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t}$.

Therefore, we conclude that distinguishing between the outputs of G_1 and G_2 is equivalent to distinguishing between the output of $\mathsf{Exp}^{\mathsf{rfnum},0}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t}$ and $\mathsf{Exp}^{\mathsf{rfnum},1}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t}$, and therefore the adversarial advantage of distinguishing between the output of G_1 and G_2 is $\delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$.

- G_3. The third game G_3 is equivalent to the election, where the voter cast a vote for a non-null voting option $v \neq 0$, and the tally result R is calculated on the bulletin board $\mathsf{BB}_{0,2}$ with simulated tally proof $\Pi = \mathsf{SimProof}(\mathsf{BB}_{0,3}, R)$.

We now prove, that the adversarial advantage in distinguishing between the output of G_2 and G_3 is negligible. Consider an adversary $\mathcal{B}$ in the ballot privacy experiment $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ who simulates the games G_2 and G_3 for the adversary $\mathcal{A}$. The adversary $\mathcal{B}$ returns the output of $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A}}$ for the queries $\mathcal{O}\mathsf{VoteLR}$, $\mathcal{O}\mathsf{Tally}$. For simulating the output of $\mathcal{O}\mathsf{Receipt}(id, v_0, v_1, t)$, $\mathcal{B}$ proceeds as follows: first, she computes a ballot $b_v = \mathsf{Vote}((id, \mathsf{sk}_{id}), id, v_0, t)$. She then chooses a random value $m \leftarrow_{\$} \mathbb{P}_d$, and a set of and random timestamps $t_1, ..., t_m \leftarrow_{\$} \mathbb{P}_t$, and computes a set of ballots $b_1, ..., b_m$ with $b_i = \mathsf{Vote}((\hat{id}, 0), id, 0, t_i)$. She then uses the query $\mathcal{O}\mathsf{VoteLR}(id, id, 0, v_1/v_0, t')$ for a random $t' \in \mathbb{P}_t$ in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A}}$ and returns its output together with the ballots $b_v, b_1, ..., b_m$ to $\mathcal{A}$. At

the end, $\mathcal{B}$ returns the value β output by $\mathcal{A}$ as the guess in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$. Thus, it follows that the adversarial advantage in distinguishing G_2 from G_3 is at most equal to the adversarial advantage in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A}}$, denoted as δ_{BPRIV}.

It follows, that in the transition through the game sequence $G_1 \rightarrow G_2 \rightarrow G_3$ the outputs of each game are distinguished from the outputs of a previous game with the advantage either $\delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t}$ (for games G_1 and G_2) or δ_{BPRIV} (for games G_2 and G_3). Hence, the adversary distinguishes between the output in $\mathsf{Exp}^{\mathsf{rfree},\beta}_{\mathcal{A}}$ with the advantage of at most $\delta^{\mathsf{rfnum}}_{\mathbb{P}_d,\mathbb{P}_t} + \delta_{BPRIV}$, with δ_{BPRIV} negligible as proven in Section 6.4.1. □

6.4.2 Fairness

Intuitively, it can be seen that fairness is implied by vote privacy in KTV-Helios, since the ballots are attached to the voters identities up until the tally. In this section we propose a following way to evaluate fairness in a formal way, whereby we show that vote privacy according to the definition of ballot privacy as described in Section 6.4.1 also implies fairness.

The idea behind our definition of fairness is as follows. The adversary has the access to the contents of the bulletin board before the voting is finished (hence, also before the tally result is published). Fairness is violated if at some point during the voting the adversary gets some information on the partial result of the election, that is, the result of tallying the ballots that have been cast so far. We model this violation as follows: given any two partial results R_0, R_1 that correspond to the same number of cast ballots k, the adversary should be unable to distinguish, which one of these results is the current partial result of the election based on the contents of the bulletin board.

Note that the adversary knows the partial results based upon the ballots of the voters that are fully under adversarial control. These ballots are therefore not considered in our definition.

Hence, in order to define fairness, we propose a following experiment $\mathsf{Exp}^{\mathsf{fairness},\beta}_{\mathcal{A},\mathcal{S}}$. The adversary selects two vectors $R_0 = (v_{0,1}, ..., v_{0,k})$, $R_1 = (v_{1,1}, ..., v_{1,k})$ with $R_0 \neq R_1$. She further selects k honest voters $(id_1, ..., id_k)$. The challenger sets up two empty bulletin boards BB_0 and BB_1, runs the setup phase as outlined in Section 6.3 and chooses a random $\beta \leftarrow_{\$} \{0,1\}$. She further computes a set of ballots $b_{i,j} = \mathsf{Vote}((id_j, \mathsf{sk}_{id_j}), id_j, v_{i,j}, t_{i,j})$ for a random timestamp $t_{i,j} \leftarrow_{\$} \mathbb{P}_t$, $i = 0, 1$, $j = 1, ..., k$ and appends each $b_{i,j}$ to BB_i. The adversary then gets to see BB_β and has to output β.

We further define fairness as follows:

Definition 6.11. *A voting scheme $\mathcal{S}$ ensures fairness, if the adversarial advantage in*

$$\mathsf{Adv}^{\mathsf{fairness}}_{\mathcal{A},\mathcal{S}} := \left| \Pr\left[\mathsf{Exp}^{\mathsf{fairness},0}_{\mathcal{A},\mathcal{S}} = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{fairness},1}_{\mathcal{A},\mathcal{S}} = 1\right] \right|$$

is negligible for any PPT adversary.

We are now ready to prove fairness for KTV-Helios.

Theorem 6.12. *The voting scheme defined in Section 6.3 provides fairness.*

Proof. We show, that an adversary in $\mathsf{Exp}^{\mathsf{fairness},\beta}_{\mathcal{A},\mathcal{S}}$ has at most the same advantage as in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$. Indeed, consider an adversary $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ who has access to an algorithm $\mathcal{B}$ that solves $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{B},\mathcal{S}}$. $\mathcal{A}$ models the experiment $\mathsf{Exp}^{\mathsf{fairness},\beta}_{\mathcal{A},\mathcal{S}}$ as follows. She returns the output of $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ in the setup to $\mathcal{B}$. Furthermore, upon getting the two result vectors R_0, R_1 from the adversary in $\mathsf{Exp}^{\mathsf{fairness},\beta}_{\mathcal{A},\mathcal{S}}$, she casts a query $\mathcal{O}\mathsf{VoteLR}(id_j, id_j, v_{1,j}, v_{0,j}, t)$ for each $j = 1, ..., k$ and returns its output by $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ to $\mathcal{B}$. After casting all k of such queries, she returns the output of $\mathcal{B}$ as her answer in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{B},\mathcal{S}}$. Hence, the adversarial advantage in $\mathsf{Exp}^{\mathsf{fairness},\beta}_{\mathcal{A},\mathcal{S}}$ is at most as large as the adversarial advantage in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$, which is negligible as shown in Section 6.4.1. Hence, fairness in KTV-Helios is ensured as long as vote privacy is ensured. □

6.4.3 Participation Privacy

In this section we evaluate the participation privacy requirement in KTV-Helios. As in the case of vote privacy including receipt-freeness, we provide formal evaluation of participation privacy given a single tabulation teller. As such, we presume that the results of our analysis should hold for the case of multiple tabulation tellers if more than half of them are honest, due to the properties of the secret sharing scheme used in generating the election key and distributed threshold decryption. We consider the formal security proofs for such case a part of future work.

We first provide a cryptographic definition of probabilistic participation privacy (Section 6.4.3). Since one may consider participation privacy an extension of vote privacy, seeing abstention as one of the possible voting options, we decided to consider modifying an existing definition of vote privacy for defining participation privacy. As such, our definition of participation privacy is inspired by the idea of vote swapping that has been used, in particular, in [BY86] to provide a game-based definition of vote privacy. The vote swapping approach considers two voters, id_0 and id_1 and two different votes v_0 and v_1, so that the adversary has to distinguish between the election where id_0 votes for v_0 and id_1 votes for v_1, or vice versa. While more advanced definitions for vote privacy have been developed (see [BCG+15] for an overview), the concepts that they use would not be suitable for defining participation privacy, since the techniques that obfuscate the content of the ballot (i.e. encryption) are generally different from the techniques that obfuscate the identities of the voters who cast their ballots. Hence, based on the vote swapping idea, we consider *voter swapping* in our definition: given two voters id_0, id_1, the adversary should be unable to distinguish whether id_0 has abstained and id_1 participated in the election, or vice versa.

Note that our definition assumes that the number of cast ballots included in the tally is revealed by the election result. On the other hand, while publishing the number of voters who participated in the election (thus, the number of ballots that were cast and included

in the tally) is often the case in practice, in both Internet voting and traditional elections, other voting systems might encode the votes in such a way, that the presentation of the final result does not reveal the number of the voters who cast their ballot. For example, given that the "yes"-vote is coded as 1 and a "no"-vote as -1, the final result presented as the sum of all the cast votes, and given that the individual ballots are not published, the result of 0 would not reveal whether there were two voters voting for 1 and -1, or no voters at all. The participation privacy for such a voting system would not be covered by our definition. However, we still consider our definition to be appropriate for KTV-Helios and other voting systems that do reveal the number of participating voters. Proposing a more general definition would be considered in future work.

In order to enable the evaluation of participation privacy in KTV-Helios, we chose to propose a quantitative definition, inspired by the coercion resistance definition in [KTV10a] and the verifiability definition in [CGKü$^+$16]. Similar to the notion of (γ_k, δ)-verifiability with quantitative goal γ_k in [CGKü$^+$16], we speak of (δ, k)-participation privacy, where δ denotes the advantage of the adversary who tries to tell whether a given voter has abstained from casting her ballot in the election, or cast her ballot at most k times. In Section 6.4.3, we instantiate this definition for the KTV-Helios and provide the optimal value of δ, so that KTV-Helios satisfies (δ, k)-participation privacy.

Defining (δ, k)-Participation Privacy: We consider the following experiment $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$ given the adversary $\mathcal{A} \in C_S$, so that C_S is a set of PPT adversaries, defined according the adversarial model for a particular scheme. There are two bulletin boards BB_0, BB_1, which are set up by the challenger. The adversary only sees the public output for one of these bulletin boards BB_β, $\beta \leftarrow_\$ \{0,1\}$. Let Q_S be a set of oracle queries which the adversary has access to. Using these queries, the adversary fills both of the bulletin boards with additional content modeling the voting, so that BB_0 and BB_1 contain the same cast ballots except for the ballots for the voters id_0, id_1: given a number of voting options $v_1, ..., v_{k'}$ chosen by the adversary, $k' \leq k$, for each $i = 0, 1$, the bulletin board BB_i contains the votes for $v_1, ..., v_{k'}$ on behalf of id_i and an abstention from the election is modeled for the voter id_{1-i}.

The oracle computes the tally result R on BB_0. In case a voting scheme provides auxiliary output Π for the tally, the oracle returns (R, Π) in case $\beta = 0$, and simulates the auxiliary output $\Pi' = \mathsf{SimProof}(\mathsf{BB}_1, R)$, returning the tuple (R, Π') in case $\beta = 1$[19]. The oracle further outputs the public content of BB_β to the adversary. The goal of the adversary is to guess whether the provided output corresponds to BB_0 or to BB_1, i.e. to guess β.

The definition of (δ, k)-participation privacy is then as follows:

[19]The tally result should be the same, if the vote of each voter is equally included in the result. However, in order to be able to model the voting schemes where the weight of the vote might depend on the voter's identity, we chose to simulate the auxiliary output in our definition.

Definition 6.13. *The voting scheme $\mathcal{S}$ achieves (δ, k)-participation privacy given a subset of PPT adversaries C_S, if for any adversary $\mathcal{A} \in C_S$, $k \in \mathbb{N}$ and two honest voter id_0, id_1 holds*

$$|\Pr\left[\mathsf{Exp}^{\mathsf{ppriv},0}_{\mathcal{A},\mathcal{S},k} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{ppriv},1}_{\mathcal{A},\mathcal{S},k} = 0\right] - \delta|$$

is negligible in the security parameter.

(δ, k)-Participation Privacy in the KTV-Helios Scheme: In order to evaluate (δ, k)-participation privacy in the KTV-Helios scheme according to the aforementioned definition, we first need to specify the adversary $\mathcal{A} \in C_S$ we aim to protect against. Afterwards we consider the information sources that would help the adversary $\mathcal{A} \in C_S$ to correctly guess β at the end of the experiment. This is done in order to determine the optimal value of δ, so that the KTV-Helios scheme satisfies (δ, k)-participation privacy for a given k with $\mathcal{A} \in C_S$ according to Theorem 6.13. We conclude the evaluation by showing how to calculate this optimal value of δ depending on the information leakage from those sources.

Specification of $\mathcal{A} \in C_S$ We make following assumptions regarding adversarial capabilities: the tabulation teller is honest, thus does not divulge the private election key to the adversary (A-KTV-TabTellerHonest), both the voting and the verification device do not leak the information to the adversary (A-KTV-VotDeviceLeakage, A-KTV-VerDeviceTrusted), the adversary is incapable of observing the communication channel between the voter, the posting trustee and the voting system (A-KTV-AnonChannels), at least one posting trustee does not divulge private information to the adversary (A-KTV-PosTrusteeHonest), the voter verifies that she communicates with an authentic bulletin board during voting, the bulletin board does not remove or modify the published data and shows the same view to everyone (A-KTV-NoBBModification, A-KTV-BBConsistency), the honest voters (aside from id_0 and id_1 in $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$) decide to participate or to abstain in the election independently from each other (A-KTV-IndAbstain) and the voters are not actively trying to prove that they abstained due to coercion (A-KTV-NoForcedAbstention). Thus, we assume that the adversary is only able to cast dummy ballots on behalf of any voter and non-dummy ballots on behalf of corrupted voters. Hence, the assumptions match the ones given in Section 6.2.

We define C_S as a set of adversaries that are given access to the queries $Q_S = \{\mathcal{O}\mathsf{Cast}, \mathcal{O}\mathsf{VoteAbstain}, \mathcal{O}\mathsf{Tally}\}$ in the experiment $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$. These queries are defined as follows:

- $\mathcal{O}\mathsf{Cast}(b)$: the adversary casts a ballot on behalf of a corrupted voter by appending b to both of the bulletin boards BB_0 and BB_1. If the ballot b is invalid (namely, $\mathsf{ValidBB}(\mathsf{BB}_\beta, b) = \perp$), the query terminates and returns $\perp$.
- $\mathcal{O}\mathsf{VoteLR}(id', v_0, v_1)$: the adversary requests an oracle to cast a ballot on behalf of an honest voter other than id_0, id_1 for either v_0 (which is appended on BB_0) or v_1 (which is appended on BB_1). If $id' \in \{id_0, id_1\}$, the query terminates and returns $\perp$.

- $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$: the oracle returns $\perp$ if $k' > k$. Otherwise, the oracle models the output of an honest posting trustee by appending a series of dummy ballots $b_1, ..., b_{m_i} \leftarrow\!\!\$\ \mathsf{VoteDummy}(id_i)$, $i = \{0, 1\}$ next to both of the voters id_0, id_1 on both of the bulletin boards BB_0 and BB_1, with m_0, m_1 sampled by the oracle according to the probability distribution $\mathbb{P}_d$ defined as in Section 6.3.Additionally, for each of the bulletin boards BB_i, $i = \{0, 1\}$, the oracle appends k' ballots $b'_j = \mathsf{Vote}((id_\beta, \mathsf{sk}_{id_i}), id_\beta, v_j, t'_j)$, $j = 1, ..., k'$ with a random timestamp $t'_j \leftarrow\!\!\$\ \mathbb{P}_t$ next to the voter id_i. The adversary is allowed to query $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$ only once.

- $\mathcal{O}\mathsf{Tally}$: The oracle returns the tally result R on BB_0 and the auxiliary data Π which is either real in case $\beta = 0$ or simulated (i.e. $\Pi = \mathsf{SimProof}(\mathsf{BB}_1, R)$) in case $\beta = 1$. The adversary is allowed to query $\mathcal{O}\mathsf{Tally}$ only once.

We now consider the sources of information that would help the adversary $\mathcal{A} \in C_S$ to correctly guess β at the end of the experiment $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$. Namely, one such source that can be used by the adversary is k' additional ballots next to id_i on the bulletin board BB_i as the output of $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$. In order to account for the adversarial advantage gained from the number of ballots next to voter's identity on the bulletin board, we define the following experiment $\mathsf{Exp}^{\mathsf{num},\beta}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k'}$: the challenger chooses a random $\beta \in 0, 1$. She then outputs two numbers m_0, m_1, so that $m_\beta = m + k'$, with $m \leftarrow\!\!\$\ \mathbb{P}_d$, and $m_{1-\beta} \leftarrow\!\!\$\ \mathbb{P}_d$. The oracle additionally returns the set of timestamps $t_{0,1}, ..., t_{m_0}, t_{m_0+1}, ..., t_{m_0+m_1}$ that are independently sampled from $\mathbb{P}_t$ to the adversary. Hence, $\beta = i$ models the election in which the voter id_{1-i} abstains and the voter id_i casts k' ballots. The adversary has to guess β. Let $\delta^{\mathsf{num}}_{k,\mathbb{P}_d,\mathbb{P}_t}$ denote an advantage in this experiment, so that $|\Pr\left[\mathsf{Exp}^{\mathsf{num},0}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{num},1}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k} = 0\right] - \delta^{\mathsf{num}}_{k,\mathbb{P}_d,\mathbb{P}_t}|$ is negligible[20]. We are now ready to evaluate (δ, k)-participation privacy, for KTV-Helios.

Theorem 6.14. *KTV-Helios, instantiated with the probability distributions $\mathbb{P}_d, \mathbb{P}_t$ achieves (δ, k)-participation privacy for a given $k > 0$ given the subset of adversaries C_S, with $\delta = \max_{k' \leq k} \delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$. It further does not achieve (δ', k)-participation privacy for any $\delta' < \delta$.*

Proof. We base our proof on the idea, that the aforementioned sources of information (i.e. the number of ballots next to id_0 and id_1) is the only ones that give advantage to the adversary. The rest of the public election data, as in case of ballot privacy (as shown in Section 6.4.1), does not provide any advantage to the adversary.

Our proof strategy is hence as follows. We consider a sequence of games, starting from $\mathsf{Exp}^{\mathsf{ppriv},0}_{\mathcal{A},\mathcal{S}k}$ and ending with $\mathsf{Exp}^{\mathsf{ppriv},1}_{\mathcal{A},\mathcal{S},k}$ and show, that the adversary $\mathcal{A}$ that is given access to the queries in Q_S distinguishes the transition through all those games with the advantage of at most $\delta := \max_{k' \leq k} \delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$. We define $\mathsf{BB}_{0,i}$ as the content of the bulletin board and

[20] We show how to calculate $\delta^{\mathsf{num}}_{k,\mathbb{P}_d,\mathbb{P}_t}$ for some choices of $\mathbb{P}_d$ and $\mathbb{P}_t$ in Section 6.6.

(R_i, Π_i) as the tally output at the end of the game G_i, $i = 1, ..., 4$. We define the sequence as follows:

- G_1. The first game G_1 is equivalent to the experiment $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$ with $\beta = 0$, and $v_1, ..., v_{k'} \neq 0$ (hence, it is equivalent to the election where the voter id_1 abstains, and the voter id_0 casts $k' \leq k$ ballots with the votes $v_1, ..., v_{k'}$). Thus, the content of $\mathsf{BB}_{0,1}$ and the tally output (R_1, Π_1) correspond to the content of BB_0 and the output of $\mathcal{O}\mathsf{Tally}$ at the end of $\mathsf{Exp}^{\mathsf{ppriv},0}_{\mathcal{A},\mathcal{S},k}$.

- G_2. The second game G_2 is equivalent to the election, where the voter id_1 abstains, and the voter id_0 casts $k' \leq k$ ballots with a null-vote each. The contents of the bulletin board $\mathsf{BB}_{0,2}$ is equivalent to the content of the bulletin board BB_0 at the end of $\mathsf{Exp}^{\mathsf{ppriv},1}_{\mathcal{A},\mathcal{S},k}$ for the adversary using the query $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$ with $v = 0$. The tally result R, however, is calculated on the contents of the bulletin board $\mathsf{BB}_{0,1}$ in the game G_1, and the auxiliary output Π_2 is simulated as $\Pi_2 = \mathsf{SimProof}(R_1, \mathsf{BB}_{0,2})$.

We prove, that the adversarial advantage in distinguishing between the output of G_1 and G_2 is at most the adversarial advantage in the ballot privacy experiment (Section 6.4.1). Consider an adversary $\mathcal{B}$ in the ballot privacy experiment $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$, who simulates the games G_1 and G_2 for the adversary $\mathcal{A}$. The adversary $\mathcal{B}$ returns the output of $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$ for the queries $\mathcal{O}\mathsf{Cast}$, $\mathcal{O}\mathsf{VoteLR}$ and $\mathcal{O}\mathsf{Tally}$. For simulating the output of $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$, $\mathcal{B}$ proceeds as follows: First, she simulates the dummy ballots for each voter id_i, $i \in \{0, 1\}$ by choosing a random values $m_i \leftarrow\!\!\$\, \mathbb{P}_d$, and a set of random timestamps $t_1, ..., t_{m_i} \leftarrow\!\!\$\, \mathbb{P}_t$. The dummy ballots $b_{i,1}, ..., b_{i,m_i}$ are computed as $b_{i,j} = \mathsf{Vote}((\hat{id}, 0), id_i, 0, t_j)$, $j = 1, ..., m_i$. Afterwards, she simulates casting the votes $v_1, ..., v_{k'}$: For each of the votes v_l, $l = 1, ..., k'$, she uses the query $\mathcal{O}\mathsf{VoteLR}(id_1, id_1, 0, v_l, t)$ for a random $t_l \in \mathbb{P}_t$ in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$. The output of the queries $\mathcal{O}\mathsf{VoteLR}$ and the dummy ballots $b_{i,1}, ..., b_{i,m_i}$ is returned to $\mathcal{A}$. At the end, $\mathcal{B}$ returns the value β output by $\mathcal{A}$ as the guess in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$. Thus, it follows that the adversarial advantage in distinguishing G_1 from G_2 is at most equal to the adversarial advantage in $\mathsf{Exp}^{\mathsf{bpriv},\beta}_{\mathcal{A},\mathcal{S}}$, denoted as δ_{BPRIV}.

- G_3. The third game G_3 is equivalent to the election, where the voter id_0 abstains, and the voter id_1 casts $k' \leq k$ ballots with null-vote each. Namely, the content of the bulletin board $\mathsf{BB}_{0,3}$ is equivalent to the content of the bulletin board BB_1 at the end of $\mathsf{Exp}^{\mathsf{ppriv},1}_{\mathcal{A},\mathcal{S},k}$ for the adversary using the query $\mathcal{O}\mathsf{VoteAbstain}(v_1, ..., v_{k'})$ with $v_l = 0$ $\forall l = 1, ..., k'$, $k' \leq k$. The tally outputs the result R_1 computed on $\mathsf{BB}_{0,1}$ and simulated auxiliary data $\Pi_3 = \mathsf{SimProof}(R2, \mathsf{BB}_{0,3})$.

We prove, that the adversary has an advantage of $\max_{k' \leq k} \delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$ of distinguishing between the output of G_2 and G_3. The tally result does not change, hence the tally output (R_1, Π_2) is equivalent to the tally output (R_1, Π_3). The only difference between the contents of $\mathsf{BB}_{0,2}$ and $\mathsf{BB}_{0,3}$ is the presence of k' additional ballots with the encryption of 0, that are published either next to id_0 (on $\mathsf{BB}_{0,2}$ in G_2) or next to id_1 (on $\mathsf{BB}_{0,3}$ in G_3) on $\mathsf{BB}_{0,3}$. Since all of these ballots encrypt 0 and due to the zero-knowledge of the attached well-formedness proofs, each individual ballot b_j, $j = 1, ..., k'$, published next to

id_i would be indistinguishable from the dummy ballots published next to id_{1-i}, $i = 0, 1$, in either one of the games G_2, G_3. Hence, the total number of the ballots next to id_0, id_1 and the timestamps of the ballots are the only source of information that can be used in distinguishing between G_2 and G_3.It follows that distinguishing between the outputs of G_2 and G_3 is equivalent to distinguishing between the output of $\mathsf{Exp}^{\mathsf{num},0}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k'}$ and $\mathsf{Exp}^{\mathsf{num},1}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k'}$ for every $k' \leq k$ chosen by the adversary, and therefore the adversarial advantage of distinguishing between the output of G_1 and G_2 is at most $\max_{k' \leq k} \delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$.

- G_4. The fourth game G_4 is equivalent to the election where the voter id_0 abstains, and the voter id_1 casts k' ballots with the votes $v_1, ..., v_{k'} \neq 0$. The tally is computed on $\mathsf{BB}_{0,1}$, and the auxiliary output is simulated as $\Pi_4 = \mathsf{SimProof}(R_1, \mathsf{BB}_{0,4})$. Applying the same argument as for the indistinguishability of G_1 and G_2, it holds that adversary distinguishes between the outputs of two games with the same advantage as in the ballot privacy experiment, namely δ_{BPRIV}.

It follows, that the in transition through the game sequence $G_1 \to G_2 \to G_3 \to G_4$, the outputs of each game are distinguished from the outputs of a previous game with the advantage either δ_{BPRIV} (for the games G_1 and G_2, and for the games G_3 and G_4) or $\delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$ for $k' \leq k$ (for the games G_1 and G_2). Since δ_{BPRIV} is negligible, as proven in Section 6.4.1, it holds that the adversary distinguishes between the output in $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},k}$ with the advantage only negligibly larger than $\delta^{\mathsf{num}}_{k,\mathbb{P}_d,\mathbb{P}_t}$ for each $k' < k$ that she chooses in the experiment. Thus, given that an adversary chooses k' so that $\delta^{\mathsf{num}}_{k,\mathbb{P}_d,\mathbb{P}_t} \geq \delta_{num,k''}\ \forall k'' \neq k', k'' \leq k$, the adversarial advantage in $\mathsf{Exp}^{\mathsf{ppriv},\beta}_{\mathcal{A},\mathcal{S},k}$ is negligibly larger than $\delta_k := \max_{k' \leq k} \delta^{\mathsf{num}}_{k',\mathbb{P}_d,\mathbb{P}_t}$. □

6.4.4 Vote Integrity and Eligibility

In order to prove the vote integrity and eligibility of the KTV-Helios scheme, we rely on the definition of weak verifiability by [CGGI14], which, under the additional assumptions that the ballots are cast as intended by the voters (i.e. not manipulated at the time of casting by the voting device, which should be ensured via corresponding verifications) and that all the voters who cast their votes verify that it appears on the bulletin board, corresponds to our definition and security model for both vote integrity and eligibility as described in Section 6.2.

Definition of Verifiability: Our goal was to prove that the scheme allows to verify that only ballots from the eligible voters, and one ballot from each voter only, are included in the tally, and that each ballot cast by eligible voters is correctly tallied. It is hence required, that a successful verification ensures, that the tally result consists of the ballots of all the honest voters who run $\mathsf{VerifyVote}(\mathsf{BB}, b)$, a subset of ballots of honest voters who did not do this, and a subset of ballots of voters corrupted by the adversary. Note, we accept the following assumptions: The bulletin board is consistent in showing the same view to everyone (A-KTV-BBConsistency). The voting register with eligible voters public signature keys is trustworthy (A-KTV-VotRegister). Furthermore, honest voters

private signature keys are not leaked to the adversary (A-KTV-VotDeviceLeakage) and the adversary is computationally restricted (A-KTV-CompRestricted). Hence, assuming that the voters perform their verifications on trustworthy verification devices (A-KTV-Audit,A-KTV-VerDeviceTrusted), the assumptions match the assumptions for vote integrity and eligibility as given in Section 6.2.

For the actual proof, we rely on the 'verifiability against a malicious bulletin board' framework definition for Helios alike schemes of [CGGI14] which we adjust the definition in [CGGI14] to the KTV-Helios scheme by applying the following experiment $\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}}$: The challenger runs the setup phase as outlined in Section 6.3 on behalf of the election organizers, the registration authority and the eligible voters. The tabulation teller, which might be controlled by the adversary, runs $\mathsf{Setup}(1^\lambda)$. The challenger further initializes an empty set I^C and HVote, which would correspond to the set of corrupted voters and to the votes cast by honest voters correspondingly. The adversary is given access to the following queries:

- $\mathcal{O}\mathsf{Cast}(b)$: appends the ballot b to the bulletin board BB.
- $\mathcal{O}\mathsf{Vote}(id', id, v, t)$: If $id \in I \cup \{\hat{id}\}$ and $id \notin I^C$, appends b to BB where $b = \mathsf{Vote}((id', \mathsf{sk}_{id'}), id, v, t)$, and adds a tuple (id', v, b) to HVote. Note, that as opposed to the definition in [CGGI14], the tuples $(id', *, *)$ already present in HVote are not removed, since the tally function takes all the valid cast ballots as the input. Otherwise, the query returns $\perp$.
- $\mathcal{O}\mathsf{Corrupt}(id)$: if called for a corrupt voter identity id $\in I^C$, the oracle immediately returns $\perp$.Otherwise, it adds id to I^C and returns the voter's private signature key sk_{id} to the adversary. In contrast to the definition in [CGGI14], we require that each tuple $(id, *, *) \in \mathsf{HVote}$ is removed from HVote, meaning that the previous ballots cast for the voter id using the $\mathcal{O}\mathsf{Vote}$ query no longer count as ballots of an honest voter.

In addition to these queries, the adversary also has the capabilities of adding, modifying and removing the ballots on the bulletin board. Additionally, a set of voters $\mathsf{Checked} \subset I$ is defined, so that for each query $\mathcal{O}\mathsf{Vote}(id, id, v, t)$, it is assumed that the corresponding voter $id \in \mathsf{Checked}$ has run $\mathsf{VerifyVote}(\mathsf{BB}, b)$ on the resulting ballot at the end of the election, and complained to the authorities in case the verification result was negative. At the end of the experiment, the adversary produces the tally output (R, Π). The experiment outputs $\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}} = 0$ if one of the following cases holds:

- There were no manipulation, i.e. the output result R corresponds to the votes from honest voters who checked that their ballot is properly stored on the bulletin board, a subset of votes from honest voters who did not perform this check, and a subset of votes from corrupted voters: i.e.

$$R = \rho((id_{E,1}, v_{E,1}), ..., (id_{E,n_E}, v_{E,n_E})) + \rho((id_{A,1}, v_{A,1}), ..., (id_{A,n_A}, v_{A,n_A})) + \rho((id_{B,1}, v_{B,1}), ..., (id_{B,n_B}, v_{B,n_B}))$$

holds; while the list of tuples $(id_{E,i}, v_{E,i})$ were cast by honest voters (i.e. $(id_{E,i}, v_{E,i}, *) \in \mathsf{HVote}$ for all $i = 1, ..., n_E$) who verified that their ballot is properly stored on the bulletin board (i.e. $id_{E,i} \in \mathsf{Checked}$ for all $i = 1, ..., n_E$); the list of tuples $\{(id_{A,1}, v_{A,1}), ..., (id_{A,n_A}, v_{A,n_A})\}$ were cast by honest voters (i.e. $(id_{A,i}, v_{A,i}, *) \in \mathsf{HVote}$ for all $i = 1, ..., n_A$) who did not verify (i.e. $id_{A,i} \notin \mathsf{Checked}$ for all $i = 1, ..., n_A$); and the list of tuples $\{(id_{B,1}, v_{B,1}), ..., (id_{B,n_B}, v_{B,n_B})\}$ represents those votes cast by the adversary so that the list $\{id_{B,1}, ..., id_{B,n_B}\}$ contains at most $|I^C|$ of unique identities (i.e. at most as many unique identities as the number of corrupted voters).

- A manipulation was detected, i.e either there were complains from the voters who run the $\mathsf{VerifyVote}$ check with $\mathsf{VerifyVote}(\mathsf{BB}, b) = \perp$, or the tally output does not pass the validity check: $\mathsf{ValidateTally}(BB, (R, \Pi)) = 0$.

The experiment $\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}}$ serves as a basis for the definition of verifiability[21] against a malicious bulletin board.

Definition 6.15. *A voting scheme $\mathcal{S}$ ensures verifiability, if the success probability in $\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}}$ $\Pr\left[\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}} = 1\right]$ is negligible for any PPT adversary.*

Proof for the KTV-Helios Scheme: We are now ready to prove the verifiability against a malicious bulletin board for the KTV-Helios scheme.

Theorem 6.16. *The voting scheme defined in Section 6.3 provides verifiability against a malicious bulletin board.*

We proceed with the proof as follows: (1) We first prove that each well-formed ballot $b_1, ..., b_n$ on the bulletin board was either cast by an honest voter who checked whether the ballot is properly stored on the bulletin board, by an honest voter who did not check this, by a corrupted voter, or the ballot corresponds to a null vote. (2) We then show that the plaintext tally result on all the votes corresponding to these ballots together correspond to the sum of a) all the votes cast by honest voters who checked that their vote is stored on the bulletin board, b) a subset of votes from honest voters who did no such checks, c) at most $|I^C|$ votes cast by the adversary and d) the dummy votes. After this, we prove (3) that this plaintext tally result corresponds to the result output by the tally function

[21] Note, that the definition can be further cast into the verifiability framework by Kuesters, Trudering and Vogt [KTV10b] in order to enable uniform treatment of verifiability. The casting of the definition by Cortier et al. [CGGI14] has been described in [CGKü+16]. Since the scheme and the security in [CGGI14] are similar to the KTV-Helios scheme and the definition of verifability used in this paper, the casting into the framework can also be done in a similar manner.

Tally, if this function is applied according to its specification in Section 6.3. We conclude by proving (4), that the adversary is incapable of producing a tally result that passes the verification check, and yet is different from the tally result output by Tally.

Step 1. Let $b = (id, c, \pi_{PoK}, \pi, t)$ be a well-formed ballot (that passes Validate) on the bulletin board. We prove that b belongs to one of the following lists with overwhelming probability:

- $\mathbb{V}^{HC}_{cast} := ((id_{E,1}, v_{E,1}), ..., (id_{E,n_E}, v_{E,n_E}))$ the list of all tuples of honest voters and non-null votes (i.e. $((id_{E,i}, v_{E,i}), *) \in \mathsf{HVote}$) who verified that their ballot is properly stored on the bulletin board (i.e. $id_{E,i} \in \mathsf{Checked}$).
- $\mathbb{V}^{HU}_{cast} := ((id_{A,1}, v_{A,1}), ..., (id_{A,n_A}, v_{A,n_A}))$, the list of all tuples of honest voters and non-null votes (i.e. $((id_{A,i}, v_{A,i}), *) \in \mathsf{HVote}$) who did not verify that their ballot is properly stored on the bulletin board (i.e. $id_{E,i} \notin \mathsf{Checked}$).
- $\mathbb{V}^{C}_{cast} := ((id_{B,1}, v_{B,1}), ..., (id_{B,n_B}, v_{B,n_B}))$, the list of all tuples of corrupted voters with non-null votes (i.e. $id_{B,i} \in I^C$) and their votes.
- $\mathbb{V}^{D}_{cast} := \{(*, 0)\}^{n_D}$: the list of all tuples that correspond to the null votes.

From the soundness of the proof π we conclude that the ciphertext c from the ballot b is signed by the voter's private signature key, or else c encrypts null, in which case b is a null-ballot and $(\hat{id}, 0)$ must be in $\mathbb{V}^{D}_{cast}$. If b is signed (i.e. it does not encrypt a null vote), by unforgeability of the signature scheme and the assumption that the private signature keys of the honest voters are not leaked to the adversary, either b was cast by a corrupt voter and so $(id, v) \in \mathbb{V}^{C}_{cast}$ where v is the vote encrypted in c, or else b was cast by a honest voter and so (id, v) must belong to one of the other two lists (depending on whether $id \in \mathsf{Checked}$ or not).

Step 2. We prove that applying the tally function ρ to the lists in step 1 outputs the tally result that includes all votes by honest voters who checked their ballots, at most I^C votes by corrupt voters and a subset of the remaining honest votes (by voters who did not check).

If there were no complaints from the voters in Checked, which would have caused the adversary to lose the security game, we know that all the ballots from these voters must be on the bulletin board so all their votes are in $\mathbb{V}^{HC}_{cast}$. The adversary's ballots are only the ones in $\mathbb{V}^{C}_{cast}$ whose identities are in I^C so the number of these ballots is at most $|I^C|$. All the remaining ballots are in $\mathbb{V}^{HU}_{cast}$ and so must have been cast by non-checking honest voters. Since ρ supports partial counting as explained in Section 6.3 we conclude, for $\mathbb{V}_{cast}$ the list of all votes in ballots on the bulletin board:

$$\rho(\mathbb{V}_{cast}) = \rho(\mathbb{V}^{HC}_{cast}) + \rho(\mathbb{V}^{HU}_{cast}) + \rho(\mathbb{V}^{C}_{cast}).$$

Step 3. We prove that applying $\mathsf{Tally}(\mathsf{BB}, \mathsf{sk})$ to the ballots on the bulletin board tallies them correctly, i.e. the result R corresponds to $\rho(\mathbb{V}_{cast})$.

The homomorphic property of ElGamal means that the ciphertexts input to the mix net contain the sum of all votes cast under the name of each voter. The mix net does not change the encrypted values in the ciphertexts, it just permutes them around. Since the ElGamal is a correct encryption scheme (i.e. the decryption outputs the correct message that was encrypted) and the PET is sound, it follows that the decrypted values output in the PET correspond to the messages in the ciphertexts output by the mix net. Hence it follows that the result output by $\mathsf{Tally}(\mathsf{BB}, \mathsf{sk})$ corresponds to the function ρ applied to the votes in the ballots on BB. (This step is essentially a proof of correctness for the KTV scheme.)

Step 4. We prove that the adversary cannot output a result/proof pair (R', Π') for a result R' different from the result R that Tally would return, which passes $\mathsf{ValidateTally}$.

The homomorphic sum-ciphertexts for each voter are recomputed by $\mathsf{ValidateTally}$ to be able to check the shuffle validity of the mix net. The mix net is protected by the proof of shuffle validity $\pi_{mix} \in \Pi'$ which is sound, so the shuffled ciphertexts $(\bar{c}_i) \in \Pi'$ must be a valid permutation and re-encryption of those on the bulletin board. The PET decryptions too are protected by a sound proof of decryption validity, so the decryption factors d in Π must match the ballots on the bulletin board. From these decryption factors, the result R can be recomputed. Therefore, unless one of the proofs in Π is invalid (which would contradict soundness) we conclude that if $\mathsf{ValidateTally}(\mathsf{BB}, (R', \Pi'))$ only outputs 1 when R is the correct result for BB.

Hence, the adversarial success probability $\Pr\left[\mathsf{Exp}^{\mathsf{ver-b}}_{\mathcal{A},\mathcal{S}} = 1\right]$ is negligible. This proves verifiability against malicious bulletin boards.

□

6.4.5 Robustness

We only provide an informal argument to the robustness requirement in our KTV-Helios. We further consider providing formal proofs as future work.

It has already been shown in previous sections of this chapter (more precisely, Section 6.4.4), that a voting system can output an election result that corresponds to the ballots published on the bulletin board, as long as the contents of the bulletin board are available for tallying, and the tabulation tellers provide valid output during the tally, namely, the results of mix net and PETs. The output of mix net can be used to ensure the privacy-related requirements of the election, as long as at least one tabulation teller is honest and performs the shuffle correctly. Furthermore, given the threshold distribution of the private election key between the tabulation tellers, it suffices if at least $t = \lfloor N_t/2 \rfloor + 1$ of the tabulation teller provide output for PETs. Hence, robustness is ensured as long as the contents of the bulletin board are available for tallying, and a threshold number of tabulation teller (that is, more than half) are available and provide valid output during tallying. We consider providing formal proofs for robustness in KTV-Helios as future work.

Note, however, that the assumption, that the contents of the bulletin board are available, can be difficult to ensure in case of *board flooding*: In the scheme in Section 6.3, as well in a family of other schemes that rely on anonymous channels for casting the ballot [CCM08, SKHS11], the adversary can exploit this possibility via a denial of service attack by casting a large amount of ballots, possibly even up to a point when bulletin board can no longer be available. Note, that the impact of such an attack, however, is considerably lower than in [CCM08], as the computations required for processing the ballots increase in linear time with the number of cast ballots, as opposed to quadratic time in [CCM08]. However, we still consider mitigating board flooding attacks in future work.

6.5 Related Work

In this chapter describe the related work on both the privacy improvements proposed in our extension (i.e. participation privacy and receipt-freeness) and formal evaluation of security requirements in Internet voting protocols.

6.5.1 Privacy Improvements

While a number of proposed Internet voting schemes provide participation privacy, most of them achieve by omitting every information identifying the voter. As such, the ballots might not be published at all (e. g. in Estonian Internet voting system [Est10]), published without any reference to the voter who cast it (e. g. in the voting approaches using blind signatures, such as [FOO92]) or published next to the pseudonyms of the voters who cast them (e. g. the newer versions of Helios [ADMP+09]). However, in all of these approaches, achieving participation privacy comes with weaker eligibility. As such, if pseudonyms are used, it is much more difficult to verify that a pseudonym (hence, a corresponding ballot) belongs to an eligible voter, than in case the actual voters identities are being published. Hence, the registration authority (i.e. the entity that assigns pseudonyms to the voters) should be trusted to ensure eligibility. Similarly, if the ballots are published with no reference to the voter, the entity who is responsible for authorizing that a ballot is included in the tally should be trusted only to accept ballots from eligible voters. Furthermore, if the cast ballots are not published at all, then both the eligibility and the vote integrity of the election cannot be verified.

An alternative approach to ensure participation privacy (referred to by the authors as anonymity) while also providing means to verify eligibility has been proposed by Haenni and Spycher [HS11]. This approach relies on a verifiable mix net shuffle of the voters DSA public signature keys, so that the anonymized keys are used for signing and publishing the cast ballots. The scheme, however, does not provide receipt-freeness, as a voter can still prove that she voted for a particular candidate by storing the randomness used in creating a ballot, just like in Helios-Base. Other approaches to ensuring participation privacy have relied on such techniques as set membership proofs [LH15] or linkable ring

signatures [HB16], without providing receipt-freeness.

A number of Internet voting schemes have been proposed that aim to ensure receipt-freeness, thus countering vote selling. Similar to ensuring participation privacy, one approach used in real-world elections for ensuring receipt-freeness consisted in refusing to publish the cast ballots, thus making it impossible for the voter to construct a receipt. Other proposals aim at ensuring receipt-freeness without hindering eligibility or vote integrity, such as the proposal by Hirt et al. [HS00] using mix net shuffle and designated-verifier proofs, proposal by Lee et al. [LBD+03] using a tamper-resistant randomizer or the extension of Helios in [CFG15] using signatures on randomizable ciphertexts. All these proposals, however, do not ensure participation privacy.

Several proposals [AKLMQ15, LH16, LHK16] aim to ensure receipt-freeness using the deniable updating approach, similar to our extension. All these proposals also ensure participation privacy. The proposal in [LH16] uses set membership proofs for ensuring participation privacy and relies on an idea of dummy ballots, similar to our extension, for receipt-freeness. However, it is significantly less efficient than our proposal, requiring e.g. $\mathcal{O}MN$ computations for verifying all the cast ballots given M as the number of eligible voters and N as the total number of cast ballots (including dummy ballots), as opposed to $\mathcal{O}(N)$ in our extension. The scheme in [LH16] furthermore does not support preliminary vote updating. Hence, the voter who waits until the last minute to cast her ballot can construct a receipt for it with a higher success probability (i.e. if no dummy ballots are cast on behalf of the voter afterwards). Our proposal, on the other hand, still allows the voters to deniably modify her vote by casting another updating ballot beforehand. The proposals in [LHK16] and [AKLMQ15] lack in efficiency as well, both requiring $\mathcal{O}(N^2)$ computations for tallying N cast ballots, as opposed to $\mathcal{O}(N)$ computations in our proposal.

The goal of ensuring both receipt-freeness and participation privacy without drawbacks in eligibility is addressed, among other security requirements, by the schemes aiming to provide the property of coercion resistance. In particular, the work of Juels, Catalano and Jacobson (JCJ) in [JCJ05] presented an Internet voting scheme that provides coercion resistance – the definition of which includes receipt freeness as well as protection against forced abstention, randomization and simulation attacks – against strong attacker. This scheme, however, is unsuited for practical use, due to the fact, that its performance is $\mathcal{O}(N^2)$ with N as the number of eligible voters. Therefore, a number of works have proposed the improvements to the JCJ system, that preserve the coercion-resistance properties while achieving linear complexity – among others, approaches based upon group signatures [AT13], panic passwords [CH11], concurrent ballot authorization [ECH12], anonymity sets [SHKS11], using the voter roll [SKHS11] or algebraic message authentication codes [ABBT16]. In particular, the work by Spycher et al. [SKHS11] also relies on the concept of dummy ballots similar to the concept used in our extension for achieving coercion resistance with linear complexity. Furthermore, several improvements focused on improving other shortcomings in JCJ scheme, such as addressing the issue of board

flooding [KHF11], improving usability with using tamper-resistant smartcards [NFVK13] or improving verifiability [Roe16] with an additional proof of knowledge during the voting. All these improvements, however, still require complex forms of credential management thus lacking in usability from the voter's perspective. A number of other schemes has been suggested that provide some level of coercion resistance [KZ07, RW11], which, however, also require complex actions from the voter. The Caveat Coercitor scheme [GRBR13] aims at detecting whether coercion took place during the election, but not at preventing it.

6.5.2 Formal Security Analysis

Significant work has been done in formal analysis of the security of Internet voting schemes. Several concrete and abstract definitions of security requirements and underlying assumptions have been developed applying various formalization approaches. As such, an overview of game-based ballot privacy definitions was proposed in [BCG$^+$15], and a framework that proposes a uniform treatment of the verifiability definitions from [Ben87, KTV10b, CGGI14, KZZ15, SFC15] is described in [CGKü$^+$16]. Other approaches for defining and evaluating the security of voting schemes include applied pi-calculus [KR05, BHM08, DKR09], process algebra [MHS12] or k-resilience terms [SVRH11]. These approaches have been applied to evaluate various voting schemes [DKR09, KRS10, SVRH11, ACW13]. In particular, the formal security analysis of Helios has been the topic of [KRS10, CGGI14, BCG$^+$15], resulting in formal proofs of its vote privacy [BCG$^+$15] as well as vote integrity and eligibility [KRS10, CGGI14]. Our work in this chapter, in particular, builds upon the results of Bernhard et al. [BCG$^+$15] for vote privacy and Cortier et al. [CGGI14] for vote integrity and eligibility.

A number of formal definitions for receipt-freeness that allow modeling and evaluating this property in the electronic voting schemes have been proposed. Kiayias et al [KZZ15] proposed a game-based definition for receipt-freeness, which ensures, that the voters do not not get any information from the voting system that can serve as a receipt. Their definition, however, considers receipt-freeness only for the voters that do not deviate from the instructions issued by the voting system. Hence, the scenarios where the voters are required to follow the instructions of the adversary in order to obtain a receipt are excluded, as opposed to our definition. Cortier et al. [CFG15, CCFG16] provide another game-based definition, which also considers the voters that can deviate from the instructions by the voting system for the sake of obtaining a receipt. Their definition, however, does not consider the so-called "counter-strategies", that would allow the voter to construct a fake receipt for the adversary, yet to vote for her desired voting option by via a specific course of action (not covered by the instructions from the voting system that only describe the voting process in absence of vote buying). Note that the receipt-freeness of KTV-Helios relies on such a counter-strategy, namely, on deniable vote updating. The simulation-based definition of Moran et al. in [MN06], as well as the definition in [KT09] based on epistemic logic, on the contrary, allow the voter to apply counter-strategies to fake her receipts. Fur-

ther symbolic definitions of receipt-freeness include [DKR09, JP06, BHM08] (see also an overview of such definitions in [Men09]), and a framework for expressing the existing definitions of receipt-freeness in the modal logics of strategic ability method has been proposed in [TJR16]. These definitions, however, are more abstract than the definition proposed in this chapter, which is game-based and tailored to the class of counter-strategies, namely, deniable vote updating, used in KTV-Helios and is game-based as the other definitions used in evaluating the security of Helios (in previous works) and KTV-Helios (in this chapter).

For now, participation privacy electronic voting has not been in the focus of research on formal security proofs. Hence, although the definitions of vote privacy (see e.g. an overview of such definitions in [BCG+15]) can be adjusted to address participation privacy, no formal definitions of this requirement have been proposed specifically. While a number of symbolic definitions of fairness was proposed (see e.g. [TMT+08, DKR09, KR05]), to the best of our knowledge, no game-based definition of fairness have been proposed yet.

6.6 Efficiency

For estimating the efficiency of our extension, we use the following approach. Let T as the number of tabulation tellers, that are responsible for both the mixing of the votes and performing the PETs with t as threshold parameter (usually suggested as $t = \lfloor T/2 \rfloor + 1$), let $N' = \sum_{i=1}^{N} m_i$ be the number of all the ballots posted during voting (including dummy ballots posted by the posting trustee), L as number of valid voting options (for example, $L = 2$ for referendum). We count the required number of modular exponentiations during each phase of the election, summarizing the findings in Table 6.1. We assume, that the verifiable re-encryption mix net scheme with the proof of shuffle validity proposed in [TW10] is used during the tallying, requiring $8N + 5$ modular exponentiations for the proof of shuffle validity and $9N + 11$ modular exponentiations for its verification. Furthermore, we assume that the DSA signatures are used for authenticating the voters.

Table 6.1: Efficiency of individual phases

Setup	$3T + t - 2 + 2L$
Voting (cast ballot, voter-side)	8
Voting (verify, server-side)	$10N'$
Tallying	$(4T + 5t - 1)NL + (19N + 16)T$

We note, that estimating N' depends on the expected value of the probability distribution $\mathbb{P}_d$. This probability distribution, however, also influences the security of the scheme, so an appropriate trade-off should be found. We show how the choice of $\mathbb{P}_d$ influences both the efficiency and either participation privacy or receipt-freeness of the scheme according to our definitions in Sections 6.4.1 and 6.4.3.

Efficiency vs. Participation Privacy: We now provide an example of how to quantify (δ, k)-participation privacy given a particular distribution for the number of dummy votes $\mathbb{P}_d$. For this we consider an adversarial advantage in the experiment $\mathsf{Exp}^{\mathsf{num},\beta}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t,k}$ defined in Section 6.4.3. Hence, we consider an adversary who only sees the number of ballots and their timestamps next to the voter.

Let $\mathbb{P}_d$ be a geometric distribution with the parameter $p \in (0, 1]$, so that the probability $\Pr[X = m] = (1-p)^m p$ for $m \geq 0$ and $\Pr[X = m] = 0$ for $m < 0$. Since the probability distribution for times of casting the dummy ballots $\mathbb{P}_t$ is chosen in such a way, that it corresponds to the distribution of times at which the voters cast their ballots, the timestamps on the ballots do not provide any additional information to the adversary. Hence, we only consider the total number of cast ballots next to the voter as the source of information for the adversary.

Let $k > 0$, $M_c \subset \mathbb{N}_0^2$ be a set of all pairs (m_0, m_1) output in $\mathsf{Exp}^{\mathsf{num},\beta}_{\mathcal{A},k}$, for which an adversary guesses $\beta = 0$ (i.e. that $m_0 = m + k$ with $m \leftarrow\!\!\$\, \mathbb{P}_d$, $m_1 \leftarrow\!\!\$\, \mathbb{P}_d$. It holds for $\delta_{num,k}$ as defined in Section 6.4.3:

$$\begin{aligned}\delta_{num,k} &:= \Pr\left[\mathsf{Exp}^{\mathsf{num},0}_{\mathcal{A},k} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{num},1}_{\mathcal{A},k} = 0\right] \\ &= \sum_{(m_0,m_1)\in M_c} \Pr[X = m_0 - k] \cdot \Pr[X = m_1] - \Pr[X = m_0] \cdot \Pr[X = m_1 - k]\end{aligned}$$

Let $M_+ := \{(m_0, m_1) \in \mathbb{N}_0^2 : P(X = m_0 - k) \cdot P(X = m_1) - P(X = m_0) \cdot P(X = m_1 - k) > 0\}$. It further holds,

$$\begin{aligned}\delta_{num,k} &\geq \sum_{(m_0,m_1)\,\in M_+} P(X = m_0 - k) \cdot \Pr[X = m_1] - \Pr[X = m_0] \cdot \Pr[X = m_1 - k] \\ &= \sum_{m_1=0}^{k-1} \sum_{m_0=k}^{\infty} \Pr[X = m_0 - k] \cdot \Pr[X = m_1] \\ &= \sum_{m_1=0}^{k-1} (1-p)^{m_1} p \sum_{m_0=0}^{\infty} (1-p)^{m_0} p \\ &= 1 - (1-p)^k\end{aligned}$$

It further follows, that an adversary who is instructed to always output $\beta = 0$ if for the output pair (m_0, m_1) if it holds that $\Pr[X = m_0 - k] \cdot \Pr[X = m_1] - \Pr[X = m_0] \cdot \Pr[X = m_1 - k] > 0$, guesses β correctly with an advantage of $1-(1-p)^k$. Hence, it holds for the adversarial advantage $\delta_{num,k} = 1-(1-p)^k$. It further holds, that $\max_{k' \leq k} \delta_{num,k'} = \delta_{num,k}$. Thus, the KTV-Helios scheme with $\mathbb{P}_{dummy}$ as a geometric distribution with parameter p achieves (δ, k)-participation privacy with $\delta = 1 - (1-p)^k$. At the same time, as an expected value of a $\mathbb{P}_t$, it holds that there would be an average of $N' = \frac{1-p}{p}$ dummy ballots for each voter.

Efficiency vs. Receipt-Freeness As in the case of participation privacy, consider $\mathbb{P}_d$ to be a geometric distribution with the parameter $p \in (0, 1]$, and $\mathbb{P}_t$ chosen to correspond to the distribution of times at which the voters cast their ballots. We consider the adversarial advantage from seeing the total number of cast ballots next to the voter (i.e. in the experiment $\mathsf{Exp}^{\mathsf{rfnum},\beta}_{\mathcal{A},\mathbb{P}_d,\mathbb{P}_t}$ described in Section 6.4.1) as follows:

Let $M_c \subset \mathbb{N}_0$ be a set of all values of m output in $\mathsf{Exp}^{\mathsf{rfnum},\beta}_{\mathcal{A},k}$, for which an adversary guesses $\beta = 0$ (i.e. that m with $m \leftarrow\!\!\$\, \mathbb{P}_d$. It holds for $\delta_{num,k}$ as defined in Section 6.4.1:

$$\begin{aligned}\delta_{num,k} &:= \Pr\left[\mathsf{Exp}^{\mathsf{num},0}_{\mathcal{A},k} = 0\right] - \Pr\left[\mathsf{Exp}^{\mathsf{num},1}_{\mathcal{A},k} = 0\right] \\ &= \sum_{m \in M_c} \Pr[X = m] - \Pr[X = m - 1]\end{aligned}$$

Let $M_+ := \{m \in \mathbb{N}_0 : P(X = m) - P(X = m - 1) > 0\}$, i.e. for the geometric distribution, $M_+ = \{0\}$. Then it holds,

$$\begin{aligned}\delta_{num,k} &\geq \sum_{m \in M_+} \Pr[X = m] - \Pr[X = m - 1] \\ &= \Pr[X = 0] \\ &= p\end{aligned}$$

Hence, an adversary who is instructed to always output $\beta = 0$ for the output $m = 0$ and $\beta = 1$ otherwise, guesses β correctly with an advantage of p. Thus, the KTV-Helios scheme with $\mathbb{P}_d$ as a geometric distribution with parameter p achieves (δ)-receipt-freeness with $\delta = p$. At the same time, as an expected value of a $\mathbb{P}_d$, it holds that there would be an average of $N' = \frac{1-p}{p}$ dummy ballots for each voter.

6.7 Application for Boardroom and Proxy Voting

In this section we briefly discuss the ways to implement the security improvements described in Chapter 6 for the settings described in Chapter 3, Chapter 4 and Chapter 5.

6.7.1 Boardroom Voting

We consider participation privacy to be a lesser issue in boardroom voting, since the information about which voters participate in the meeting is most likely public. However, the security of boardroom voting can still be improved by ensuring receipt-freeness.

Note, however, that the receipt-freeness of our extension is ensured due to deniable vote updating, which in turn is possible only if a voter is capable of updating her vote without being observed by the adversary. Hence, it is difficult to ensure receipt-freeness for the voters who are physically present in the same room. We therefore focus on ensuring receipt-freeness for the voters who vote remotely (given that an adversary is not observing them during the voting) or against an external adversary for the voters who vote from the same location.

For modifying the scheme from Chapter 6 we introduce the following change to the voting process:

The scheme in Chapter 6 requires either an honest bulletin board or an anonymous channel between the bulletin board and the voter or the posting trustee for preserving the participation privacy and receipt freeness. The absence of a centralised bulletin board in boardroom voting, however, requires finding another solution for ensuring, that a dummy ballot is indistinguishable from a ballot that is meant to deniably update the voter's vote. One possible way to address this challenge would be to rely on existing solutions that ensure anonymous communications, such as onion routing. In the absence of a centralized broadcast channel, however, these solutions, require a significant overhead in both required computations and communicated data for each transferred message. Due to the efficiency constrains in boardroom voting, such an overhead might be unsuitable for such a setting.

Hence, we propose a modification to the security model of the scheme. All the voters are required to take over the task of a posting trustee by casting a random number of dummy ballots on behalf of all the other voters. Whenever a voter wants to deniably cast a ballot, instead of broadcasting it directly, she sends it via a private channel to another voter whom she trusts. This voter, in turn, replaces one of her dummy ballots she wanted to cast on behalf of the sender.

The modified scheme for boardroom voting would run as follows. As in the boardroom voting scheme from Chapter 3, for the setup the voters run a decentralised key exchange in order to exchange their public signature keys and generate symmetric secret keys for private communication. The voting then consists of two rounds. In the first round, the voters cast their ballots as described in Chapter 3. If a voter V_c wants to fake a receipt and deniably update her vote for v_A for a vote for v_B, she computes her ballot b_v encrypting $v_A - v_B$ as described in Section 6.3 and sends it privately to another trusted voter V_t.

In the second round, each voter V_i randomly chooses $N-1$ values $m_{i,j} \leftarrow\!\!\$\, \mathbb{P}_t$, $j = 1, ..., N$, $j \neq i$ and casts $m_{i,j}$ dummy ballots for each voter V_j[22]. If a voter V_t has got a ballot b_v from the voter V_c in the previous round, instead of one of the $m_{t,k}$ dummy ballots for V_i she casts the ballot b_v. After the voting period has ended, the votes are being tallied according to Section 6.3, with voters taking over the tasks of the tabulation tellers.

6.7.2 Proxy Voting

The extent, to which participation privacy and receipt-freeness should extend to the proxy voting setting, is a topic for further discussion. As such, we consider following variants for defining participation privacy in context of proxy voting:

Direct Voting vs. Abstaining. The adversary is unable to tell whether the voter cast a direct ballot or abstained from the election.

[22] As this step can be performed in the back-end by the voters voting devices, it does not influence the usability of the voting process

Direct Voting vs. Delegating. The adversary is unable to tell whether the voter cast a direct ballot or delegated to a proxy.

Delegating vs. Cancelling. The adversary is unable to tell whether the voter canceled their delegation by casting a direct ballot.

Delegating vs. Abstaining. The adversary is unable to tell whether the voter delegated to a proxy or abstained from the election.

Participation Privacy for Proxies. The adversary is unable to tell whether the proxy cast a delegated ballot.

Accordingly, we can distinguish between following variants of receipt-freeness:

Direct Voting. The voter is unable to construct a receipt for casting a direct ballot for a particular voting option.

Delegating. The voter is unable to construct a receipt for delegating to a particular proxy.

Casting a Delegated Ballot. The proxy is unable to construct a receipt for casting a delegated ballot for a particular voting option.

Note, that an extension proposed in Chapter 4 already ensures participation privacy for proxies, given an anonymous channel between the proxies and the bulletin board. Hence, we no longer consider it in further discussions.

We further propose two solutions to introduce participation privacy and receipt-freeness to proxy voting. Our first solution, referred to as *weaker participation privacy and receipt-freeness*, ensures participation privacy with regards to direct voting vs. abstaining and delegating vs. canceling and receipt-freeness for direct voting and delegating. and to a certain extent casting a delegated ballot. The second solution, referred to as *stronger participation privacy and receipt-freeness*, also ensures participation privacy with regards to direct voting vs. delegating and delegating vs. abstaining and receipt-freeness for casting a delegated ballot. Both of the solutions are further described.

Weaker Participation Privacy and Receipt-Freeness

Our first solution uses the deniable vote updating principle from the scheme from Chapter 6 to ensure that the voter is unable to prove that her direct ballot was cast for a particular voting option. The same principle ensures, that even if the voter is forced to delegate her vote to a malicious proxy, she can make it invalid and cancel the delegation with a direct ballot without the adversary noticing. The proxy, on the other hand, can prove to the adversary how the cast her delegated ballots, but she cannot prove that these ballots would be included into the tally (i.e. they are attached to valid delegation credentials and not overwritten by direct ballot).

We consider a list of valid voting options $\mathbb{V} \subset \mathbb{G}_q$ and a predetermined value $d \in \mathbb{G}_q \setminus \mathbb{V} \cup \{0\}$ that indicates that the voter chose to delegate her vote. We furthermore require a function $f : \mathbb{G}_q^4 \to \mathbb{G}_q$ that, given two ciphertexts $c, c' \in \mathbb{G}_q^2$ produces following output without revealing further information about the plaintexts encrypted in c_1, c_2:

$$f(c, c') = \begin{cases} \mathsf{Dec}(c) & \text{if } \mathsf{Dec}(c) \in \mathbb{V} \\ \mathsf{Dec}(c) & \text{if} \mathsf{Dec}(c) = d \text{ and } \mathsf{Dec}(c') \in \mathbb{V} \\ 0 & \text{otherwise} \end{cases}$$

This function can be implemented via PETs calculated by the tabulation tellers. Its purpose is to filter the ballots in the following way: given c_1 as the encryption of a voting option in a direct ballot next to the voter's identity and c_2 as the encryption of a voting option in a delegated ballot from the same voter, f outputs the plaintext of c_1 if it is a valid vote, the plaintext of c_2 if the voter indicated that she delegates by casting d and the corresponding proxy casts a ballot for a valid voting option, and a null vote in all the other cases.

Our modified scheme can then be described as follows. For the sake of simplicity, we only describe the voters having one delegation priority.

The setup runs as described in Chapter 6 and Chapter 4, resulting in the publication of the voting register with the voters public signature keys and of the voters delegation credentials. During the voting, the voters who choose to cast a direct ballot (or to cancel their delegation) cast their ballots as described in Chapter 6, and the posting trustee likewise casts dummy ballots on behalf of each voter. Delegating occurs as described in Chapter 4. During the tallying, the ciphertexts next to each voter that represent her direct ballot are multiplied together, the same way as in *Chapter* 6, forming a list of ciphertexts $c_1, ..., c_N$. Correspondingly, as in Chapter 4, the delegated ballots cast by proxies are processed by the tabulation tellers: the ballots are anonymized via mix net, the delegation credentials attached to the ballots are decrypted after the anonymization and the ballots are assigned according to these credentials to the corresponding voters. As a result, for the voters $id_1, ..., id_N$ a list of ciphertexts $c'_1, ..., c'_N$ is formed, whereby c'_i denotes an encrypted voting option cast in a delegated ballot for the voter id_i, and $c'_i = \mathsf{Enc}(0)$ for the voters, on which behalf no delegated ballots have been cast. The tuples (c_i, c'_i) are further anonymized via shuffling, and afterwards the function f is applied in order to assign each tuple to either a valid voting option or a null vote.

Stronger Participation Privacy and Receipt-Freeness

Our second solution, in addition to the dummy ballots for direct ballots, it also introduces dummy ballots for the delegated ballots. These dummy ballots are constructed in the following way:

Recall, that in order to construct a delegation token in Chapter 4, the voter id_i has to submit her encrypted delegation credential $(a_d, b_d) = \mathsf{Enc} g^{x_i}$ with g^{x_i} as her public part of

the delegation credential and x_i as a secret key[23], and the following proof as her delegation token:

$$\pi_d = PoK\{(r_d, x_i) : a_d = g^{r_d} \wedge b_d = g^{x_i} h^{r_d}\}(\sigma)$$

The proxy then proves the knowledge of $m = \log_g \sigma$ that she got from the voter via a private channel. During the tallying, (a_d, b_d) is decrypted to reveal a delegation credential g_{x_i}.

We modify this proof to enable casting dummy ballots for delegating votes. Namely, the proof should enable the posting trustee to construct dummy delegation tokens with the credentials g_{x_i} without knowing x_i; however, these dummy tokens should only enable casting null votes. For this purpose, we require an independent generator $\hat{g}$, so that $\log_g \hat{g}$ is unknown. The delegation token on behalf of the voter id_i (computed either by the voter herself or by the posting trustee) consists of the following values:

- a ciphertext $(a_d, b_d) = \mathsf{Enc} g^{x_i}$,
- a value $\sigma = \hat{g}^m$ or $\sigma = g^m$ for a random m,
- a proof of knowledge π_d as

$$\pi_d = PoK\{(r_d, x_i, m) : \sigma = \hat{g}^m \vee a_d = g^{r_d} \wedge b_d = g^{x_i} h^{r_d} \wedge \sigma = g^m\}(\sigma)$$

Thus, given that only the voter knows the value of x_i, only she can cast delegation tokens using $\sigma = g^m$, while the posting trustee has to set $\sigma = \hat{g}^m$ in constructing her tokens. For casting a delegated ballot (a_v, b_v) corresponding to the delegation token $((a_d, b_d)\sigma, \pi_d)$, then, the proxy or the posting trustee computes a proof of knowledge:

$$\pi_v = PoK\{(r_v, m) : a_v = g^{r_v} \wedge b_v = h^{r_v} \vee \sigma = g^m\}$$

Thus, since $g, \hat{g}$ are independent generators, and for a given σ one can only know the value of either $\log_g \sigma$ or $\log_{\hat{g}} \sigma$, it follows that the dummy delegation tokens can be only used to cast null votes.

Given this method for constructing the dummy ballots, the tallying proceeds in the same way as described above, except that the delegated ballot cast using the same delegation credential (which include dummy delegated ballots) are multiplied together, similarly to the direct ballots. In this way, participation privacy with regards to direct voting vs. delegating and delegating vs. abstaining is ensured in that the adversary is unable to tell, whether the voter has delegated her vote in the election.

[23] Again, for the sake of simplicity, we only describe delegating with one priority, hence the index j is omitted.

6.7.3 Proxy Boardroom Voting

As in Section 6.7.1, we consider participation privacy to be of less relevance to the boardroom voting setting, which extends to the boardroom functionality as well. Hence, we describe the ways to use the techniques in Chapter 6 to add receipt-freeness to the scheme described in Chapter 5.

For direct ballots, we propose to use dummy ballots as described in Section 6.7.1. Namely, the dummy ballots on behalf of each voter should be cast by other voters. In case the voter wants to update her vote, she sends her new ballot to another trusted voter, who in turn casts it in place of a dummy ballot.

For ensuring receipt-freeness for delegated ballots, we first describe the possible ways to prevent the voters from constructing receipts that they delegated to a particular proxy. Note that the scheme proposed in Chapter 5 already ensures such a property. Recall, that in this scheme each voter id_i possesses a pair of keys g_i, h_i, so that only the voter knows the value of $x_i = \log_{g_i} h_i$. In order to delegate her vote, the voter id_i computes the shares of a random secret value m_i as $m_{i,j}$ using Shamir secret sharing (Section 2.2.10) and sending the value of $g_i^{m_{i,j}}$ to the proxy id_j together with the signature on commitments $c_{i,j} = (c_{i,j}^{(1)}, c_{i,j}^{(2)})$ with $c_{i,j}^{(1)} = g_i^{r_{i,j}} h_i^{u_{i,j}}$, $c_{i,j}^{(2)} = g_i^{m_{i,j}} h_i^{r_{i,j}}$ for random $r_{i,j}, u_{i,j} \in \mathbb{Z}_q$ to each proxy id_j. She then sends the value m_i to her chosen proxy, and a random value $m'_{i,j}$ to every other proxy. Given that m_i cannot be reconstructed unless more than half of the proxies reveal their shares $g_i^{m_{i,j}}$, delegation privacy is preserved, so that only the voter knows the identity of her chosen proxy.

It holds that the voter herself cannot prove that she delegated to a specific proxy to a third party, even if she reveals all the values m_i, $m_{i,j}$, $m'_{i,j}$, $r_{i,j}, u_{i,j}$ generated during the delegation. Consider the situation, where the voter attempts to convince an adversary that she delegated to the proxy id_k, while delegating to another proxy. It holds, that $m\prime i, k$ is a random value, while the shares $m_{i,j}$ reconstruct to another value m_i instead with $m_i \neq m'_{i,k}$. In order to fake her receipt, the voter has to compute a new set of shares $m\prime i, j$ that reconstruct to $m'_{i,k}$. Given N_p as the total number of proxies and t as the threshold for the shares $m_{i,j}$, the voter has to replace at least $N_p - t + 1$ shares $m_{i,j}$ with $m''_{i,j}$. The voter then finds the corresponding values $r''_{i,j}$, $u''_{i,j}$, so that computing the commitments given the values the values $m\prime i, j, r''_{i,j}, u''_{i,j}$ results in the same values $c_{i,j} = (c_{i,j}^{(1)}, c_{i,j}^{(2)})$. Namely, it should hold, $c_{i,j}^{(1)} = g_i^{r''_{i,j}} h_i^{u''_{i,j}}$, $c_{i,j}^{(2)} = g_i^{m''_{i,j}} h_i^{r''_{i,j}}$. Given that the voter knows the value of x_i with $h_i = g^{x_i}$, she can compute such $r''_{i,j}, u''_{i,j}$, so that $x_i r_{i,j} + m_{i,j} = x_i r'_{i,j} + m'_{i,j}$ and $x_i u_{i,j} + r_{i,j} = x_i u'_{i,j} + r'_{i,j}$. Hence, as long as the adversary does not get access to the real shares $m_{i,j}$ that were replaced by the voter with $m''_{i,j}$ (i.e. as long as at least t proxies are honest), she would not be able to distinguish between the fake receipt and the real values m_i, $m_{i,j}$, $m'_{i,j}$, $r_{i,j}, u_{i,j}$.

We furthermore consider receipt-freeness for proxies by allowing the proxies to cast dummy ballots for delegated ballots as well. Recall, that for casting a delegated ballot on behalf of the voter id_i, the proxy id_j broadcasts a tuple $(\hat{e}_{i,j}^{(d)}, E_{i,j}^{(d)}, \pi_{i,j})$ with $\hat{e}_{i,j}^{(d)}$ as an

encryption of the value $g_i^{m'_{i,j}}$ as received from the voter (which is the shared value $g_i^{m_i}$ if id_j is the proxy chosen by id_i or a random value otherwise), $E_{i,j}^{(d)}$ as an encryption of a chosen voting option, and $\pi_{i,j}$ as the proof of knowledge of plaintext discrete logarithm $m'_{i,j}$ for $\hat{e}_{i,j}^{(d)}$. Note, that at this point in the scheme, the value $e_i^{(d)}$ which is an encryption of $g_i^{m_i}$ is available to all the proxies; hence, the proof of knowledge of plaintext discrete logarithm ensures, that the proxy does not simply submit a re-encryption of $e_i^{(d)}$.

The proof of knowledge of plaintext discrete logarithm, however, could be replaced by a disjunctive proof that either the proxy knows the plaintext discrete logarithm of $\hat{e}_{i,j}^{(d)}$, or she is casting a null vote, encrypted in $E_{i,j}^{(d)}$. In this way, if she wants to casts a dummy delegated ballot, she calculates $\hat{e}_{i,j}^{(d)}$ as a re-encryption of $e_i^{(d)}$. The dummy delegated ballots are then cast by other proxies on behalf of each delegating voter id_i and each proxy id_j, while the proxy that wants to deniably update her delegating vote does so by sending her new ballot to another trusted proxy, who in turn casts it instead of a dummy ballot. Afterwards, for each delegating voter id_i and proxy id_j, all the encrypted votes cast on behalf of id_i, id_j are multiplied together and processed further as described in Chapter 5, with PETs replacing distributed threshold decryption at the end.

6.8 Summary and Future Work

The final contribution of this thesis extends Helios-Base by introducing several privacy improvements. First, this extension preserves participation privacy by hiding which voters has cast their vote in the election, while at the same time providing means to verify that only the votes from eligible voters are included in the tally. The second privacy improvement introduces receipt-freeness by preventing the voters constructing receipts that prove to a third party how they voted.

6.8.1 Summary

The main idea behind our extension that allows preserving participation privacy and receipt-freeness, as well as the security requirements such as vote privacy, vote integrity and eligibility from the original Helios-Base, is the introduction of dummy ballots. A random number of the dummy ballots is cast by a new kind of entity, the posting trustee, on behalf of each voter. Hence, by obfuscating the real ballots cast by the voters themselves, the dummy ballots ensure participation privacy. Receipt-freeness is ensured due to deniable vote updating, as the voter can change their vote by casting a new ballot, while the presence of dummy ballots allows her to deny doing so.

We have formally evaluated the security of the proposed extension. Namely, we have proven that it satisfies vote privacy, vote integrity and eligibility by relying the definitions from [BCG+15, CGGI14] adjusted to the context of our extension. Furthermore, we proposed a probabilistic abstract definition of (δ, k)-participation privacy, with δ representing

the adversarial advantage in distinguishing whether a particular honest voter has cast up to k ballots in the election. We also proposed a probabilistic abstract definition of δ-receipt-freeness for voting schemes based on deniable vote updating. We proposed instantiations of both (δ, k)-participation privacy and δ-receipt-freeness definitions for KTV-Helios and determined the value of δ as the adversarial advantage for both of these properties. We furthermore proposed a formal definition of fairness followed by the security evaluation of KTV-Helios, and conducted an informal evaluation for robustness.

We furthermore discussed the issue of extending the technique of dummy ballots to improve the security of other extensions that we proposed in this work. As such, we proposed the way to adjust the scheme in Chapter 3 to ensure receipt-freeness in boardroom voting setting. We also described the adjustments to the schemes in Chapter 4 and Chapter 5 to ensure participation privacy and receipt-freeness in the context of proxy voting and proxy boardroom voting for voters casting a direct ballot, as well as for the voters delegating to a proxy.

6.8.2 Future Work

While we have provided formal proofs for our extension in the thesis, several of these proofs assumed that both the tabulation teller and the posting trustee are implemented as a single entity each. Our extension, however, also supports implementing both of these entities in a distributed way, whereby multiple posting trustees cast dummy ballots independently from each other, and multiple tabulation tellers jointly generate the election key via distributed threshold secret sharing and perform the tally with distributed threshold decryption. Although we provided an informal argument that the results of the formal proofs remain valid for multiple entities as well, it would be useful to provide more rigorous formal proofs for this claim.

Furthermore, as we only briefly discussed the proposals to introduce the privacy improvements of participation privacy and receipt-freeness in our extensions proposed in Chapters 3 to 5, one direction of future work would be to conduct a more detailed security analysis for our proposals.

Other directions of future work can focus on the further security improvements of the scheme. As such, as mentioned in Section 6.4.5, the scheme in Section 6.3 is vulnerable to board flooding, that considerably hinders the efficiency of the election, or even stop the tally from being computed. Extensions designed to solve this problem for other Internet voting schemes with anonymous channel for voting has been proposed [KHF11, HK13], however, it alters the adversarial model by requiring additional trust assumptions and probabilistically increasing the adversarial advantage.

Another issue that can be improved lies in the trusted platform problem. Helios-Base offers the method of verification to ensure that their vote has been encrypted correctly using a second device. This method, however, only offers probabilistic assurance, and requires a series of complex actions from the voter, thus substantially lowering the usabil-

ity of the voting process. Furthermore, in KTV-Helios, the vote integrity requirement is further hindered in case of malicious voting device, in that it can cast ballots on behalf of the voter without being noticed, thus altering the vote that is included in the tallying. Note, that this vulnerability is also present in other schemes that rely on deniable voter updating [LH16,LHK16,AKLMQ15] or on the voter being able to cast a vote unnoticed to the adversary [CCM08,SKHS11]. One way to solve this could be integrating the solution of Guasch et al. [GM16] for improving the vote integrity in Helios. This approach involves two devices that are used for casting a ballot, together with the designated-verifier proofs. Applying it to our scheme would improve the vote integrity assurances, in that the adversary now has to corrupt both devices for manipulating the ballot during voting. However, the security requirements related to privacy (vote privacy including receipt-freeness, fairness and participation privacy) suffer: in case at least one of the devices is corrupted, all of these requirements would be broken.

A possible solution to both the trusted platform and board flooding problems could be integrating the idea of *posting tokens*, that are required for casting each ballot, while distributing them via non-electronic channel. As such, they might be distributed via post, each token printed on a separate paper as a QR code, and the voter required to scan the code each time she wants to cast a ballot. In order to preserve the security of the scheme, however, the link between the voter and the assigned tokens has to be private, as well as the number of tokens each voter has received. For ensuring this, the methods described by Haenni et al in [HK13]. can be applied as adjusted towards paper-based tokens; however, the issue of the resulting adversarial model still has to be considered.

Aside from the security issues, one also has to consider the challenges in usability and understandability of the proposed scheme. For example, the voters might get confused seeing several ballots cast next to their identity, thus leading to distrust in the system. It is also probable that the need to remember the plaintexts of all the previously cast ballots in order to be able to update them would become an issue. Many of the complexities of the protocol could be hidden behind a helpful user interface, for example one that remembered what votes had been cast before. Nevertheless the trade-offs between security, public understanding, and ease of use remain challenging, and require further exploration (for example, in forms of user studies).

Chapter 7

Conclusion

The Helios voting system has been originally proposed by Adida in 2008 [Adi08]. Helios has since then been widely studied in the literature and used for many real-world elections, including the IACR annual internal elections [IAC16]. However, there are election settings where Helios cannot be employed, such as the four election settings addressed in this thesis. For each one of those settings we proposed a corresponding extension to Helios, more precisely, to Helios-Base, a modified version that incorporates some of its later developments and security extensions is used (see Section 2.3.3).

The ideas employed in designing the extensions can also be used in order to extend other Internet voting schemes. As such, the constructions of delegation tokens introduced in Chapter 4 and Chapter 5 can be used to enable proxy voting in other schemes that rely on the security models different from Helios, such as the JCJ scheme [JCJ05] that enables coercion resistance. The concept of dummy ballots introduced in Chapter 6 can be used to introduce participation privacy and receipt-freeness both in the other schemes described in this thesis in Chapters 3 to 5 and in other schemes, such as the ones used in the Internet voting systems in practice (e. g. in Estonia [HW14] or in Switzerland [GGP15]). Hence, the concepts described in the thesis can further contribute to opening additional possibilities to conduct secure Internet voting in further election settings.

Another contribution of this thesis are the new formal security definitions for participation privacy, fairness and receipt-freeness provided in Chapter 6 and used for formally proving the security of the extension proposed in this chapter. These definitions can also be used to provide formal security proofs for other schemes. Hence, in addition to potentially enabling Internet voting in additional settings, the contributions of this thesis can be employed to achieve more confidence in the security claims of Internet voting schemes.

Bibliography

[ABBT16] Roberto Araújo, Amira Barki, Solenn Brunet, and Jacques Traoré. Remote Electronic Voting can be Efficient, Verifiable and Coercion-Resistant. In *FC 2016: 1st Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2016*. Springer, February 2016. Cited on pages 114 and 115.

[ACW13] Mathilde Arnaud, Véronique Cortier, and Cyrille Wiedling. Analysis of an electronic boardroom voting system. In *VoteID 2013: 4th International Conference on E-Voting and Identify*, pages 109–126. Springer, July 2013. Cited on pages 37 and 115.

[Adi08] Ben Adida. Helios: Web-based open-audit voting. In *SS 2008: 17th Conference on Security Symposium*, pages 335–348. USENIX, July 2008. Cited on pages 1, 16, 20, and 127.

[ADMP+09] Ben Adida, Olivier De Marneffe, Olivier Pereira, Jean-Jacques Quisquater, and others. Electing a university president using open-audit voting: Analysis of real-world use of Helios. *EVT/VOTE 2009: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*, 9:10–10, 2009. Cited on pages 1, 2, 5, 16, 18, 20, 21, 113, and 17.

[AKLMQ15] Dirk Achenbach, Carmen Kempka, Bernhard Löwe, and Jörn Müller-Quade. Improved coercion-resistant electronic elections through deniable re-voting. *JETS: USENIX Journal of Election Technology and Systems*, 3(2):26–45, 2015. Cited on pages 97, 114, and 126.

[AKV05] Ammar Alkassar, Robert Krimmer, and Melanie Volkamer. Online-Wahlen f ür Gremien. *DuD Datenschutz und Datensicherheit*, 8(29), 2005. Cited on page 36.

[ASW98] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. In *EUROCRYPT 1998: 17th International Conference on the Theory and Application of Cryptographic Techniques*, pages 591–606. Springer, June 1998. Cited on page 12.

[AT13] Roberto Araújo and Jacques Traoré. A practical coercion resistant voting scheme revisited. In *VoteID 2013: 4th International Conference on E-Voting and Identify*, pages 193–209. Springer, July 2013. Cited on pages 114 and 115.

[BCG+15] D. Bernhard, V. Cortier, D. Galindo, O. Pereira, and B. Warinschi. Sok: A comprehensive analysis of game-based ballot privacy definitions. In *SP 2015: 36th IEEE Symposium on Security and Privacy*, pages 499–516. IEEE, May 2015. Cited on pages 2, 7, 16, 23, 83, 86, 89, 90, 94, 95, 103, 115, 116, 124, and 125.

[Ben87] Josh Daniel Cohen Benaloh. Verifiable secret-ballot elections. *PhD thesis, Yale University, Department of Computer Science*, 1987. Cited on page 115.

[Ben06] Josh Benaloh. Simple verifiable elections. In *EVT 2006: Electronic Voting Technology Workshop*, pages 5–5. USENIX, 2006. Cited on pages 16, 23, and 22.

[BG12] Stephanie Bayer and Jens Groth. Efficient zero-knowledge argument for correctness of a shuffle. In *EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 263–280. Springer, April 2012. Cited on pages 13 and 14.

[BGRR13] Katharina Bräunlich, Rüdiger Grimm, Philipp Richter, and Alexander Roßnagel. *Sichere Internetwahlen*. Nomos Verlagsgesellschaft mbH & Co. KG, 2013. Cited on page 7.

[BHM08] M. Backes, C. Hritcu, and M. Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *CSFS 2008: 21st IEEE Computer Security Foundations Symposium*, pages 195–209. IEEE, June 2008. Cited on pages 115 and 116.

[BHPS16] Michael Backes, Christian Hammer, David Pfaff, and Malte Skoruppa. Implementation-level analysis of the JavaScript Helios voting client. In *SAC 2016: 31st Annual ACM Symposium on Applied Computing*, pages 2071–2078. ACM, April 2016. Cited on page 2.

[BKV16] David Bernhard, Oksana Kulyk, and Melanie Volkamer. Security proofs for participation privacy, receipt-freeness, ballot privacy, and verifiability against malicious bulletin board for the helios voting scheme. Cryptology ePrint Archive, Report 2016/431, May 2016. `http://eprint.iacr.org/2016/431`. Cited on page 78.

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to Helios. In *ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security*, pages 626–643. Springer, December 2012. Cited on pages 2, 10, 11, 16, 17, 18, 20, 23, and 83.

[BY86] Josh C Benaloh and Moti Yung. Distributing the power of a government to enhance the privacy of voters. In *PODC 1986: 5th annual ACM symposium on Principles of distributed computing*, pages 52–62. ACM, August 1986. Cited on page 103.

[CBP16] Daniel Chung, Matt Bishop, and Sean Peisert. Distributed helios-mitigating denial of service attacks in online voting. UC Davis: College of Engineering. Retrieved from: https://escholarship.org/uc/item/7xs630v9, 2016. Cited on page 18.

[CCFG16] Pyrros Chaidos, Véronique Cortier, Georg Fuchsbauer, and David Galindo. Beleniosrf: A non-interactive receipt-free electronic voting scheme. In *CCS 2016: 23st ACM SIGSAC Conference on Computer and Communications Security*, pages 1614–1625. ACM, October 2016. Cited on pages 98 and 115.

[CCM08] Michael R. Clarkson, Stephen Chong, and Andrew C. Myers. Civitas: Toward a secure voting system. In *SP 2008: 29th IEEE Symposium on Security and Privacy*, pages 354–368. IEEE, May 2008. Cited on pages 57, 60, 113, and 126.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO 1994: 14th Annual International Cryptology Conference on Advances in Cryptology*, pages 174–187. Springer, August 1994. Cited on pages 10 and 11.

[CFE16] Nicholas Chang-Fong and Aleksander Essex. The cloudier side of cryptographic end-to-end verifiable voting: a security analysis of Helios. In *ACSAC 2016: 32nd Annual Conference on Computer Security Applications*, pages 324–335. ACM, December 2016. Cited on page 2.

[CFG15] Véronique Cortier, Georg Fuchsbauer, and David Galindo. Beleniosrf: A strongly receipt-free electronic voting scheme. Cryptology ePrint Archive, Report 2015/629, June 2015. `http://eprint.iacr.org/`. Cited on pages 17, 98, 114, and 115.

[CGGI13] Véronique Cortier, David Galindo, Stéphane Glondu, and Malika Izabachène. Distributed elgamal à la pedersen: Application to Helios. In *WPES 2013: 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, pages 131–142. ACM, November 2013. Cited on pages 13, 18, 19, 21, 23, 37, 17, and 20.

[CGGI14] Véronique Cortier, David Galindo, Stéphane Glondu, and Malika Izabachène. Election verifiability for Helios under weaker trust assumptions. In *ESORICS 2014: 19th European Symposium on Research in Computer Security, Part II*, pages 327–344. Springer, September 2014. Cited on pages 2, 8, 16, 17, 18, 19, 20, 23, 85, 108, 109, 110, 115, 124, and 125.

[CGKü+16] Veronique Cortier, David Galindo, Ralf K üsters, Johannes Mueller, and Tomasz Truderung. Verifiability notions for e-voting protocols, March 2016. Cryptology ePrint Archive, Report 2016/287 http://eprint.iacr.org/. Cited on pages 7, 104, 110, and 115.

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. *European transactions on Telecommunications*, 8(5):481–490, 1997. Cited on pages 18, 37, and 17.

[CH11] Jeremy Clark and Urs Hengartner. Selections: Internet voting with over-the-shoulder coercion-resistance. In *FC 2011: 15th international conference on Financial Cryptography and Data Security*, pages 47–61. Springer, February 2011. Cited on pages 114 and 115.

[Cou04] Council of Europe. Legal, operational and technical standards for e-voting. recommendation rec(2004)11 adopted by the committee of ministers of the council of europe on 30 september 2004 and explanatory memorandum. Council of Europe publishing, 2004. Cited on page 7.

[CP92] David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *CRYPTO 1992: 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 89–105. Springer, August 1992. Cited on pages 10, 11, 83, and 9.

[CS97a] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups. In *CRYPTO 1997: 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 410–424. Springer, August 1997. Cited on pages 10 and 9.

[CS97b] Jan Camenisch and Markus Stadler. Proof systems for general statements about discrete logarithms. Technical report, Citeseer, 1997. Cited on pages 10 and 68.

[DC12] Yvo Desmedt and Pyrros Chaidos. Applying divertibility to blind ballot copying in the helios internet voting system. In *ESORICS 2012: 17th European Symposium on Research in Computer Security*, pages 433–450. Springer, September 2012. Cited on pages 18 and 59.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976. Cited on page 15.

[DJV12] Denise Demirel, Hugo Jonker, and and Melanie Volkamer. Random block verification: Improving the Norwegian electoral mix-net. In *EVOTE 2012: 5th International Conference on Electronic Voting*, pages 65–78. Gesellschaft für Informatik, July 2012. Cited on pages 13 and 14.

[DKR09] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, 2009. Cited on pages 7, 8, 115, and 116.

[DLM82] Richard A DeMillo, Nancy A Lynch, and Michael J Merritt. Cryptographic protocols. In *STOC 1982: 14th annual ACM symposium on Theory of computing*, pages 383–400. ACM, May 1982. Cited on page 36.

[DMS04] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004. Cited on page 47.

[DVDGdSA12] Denise Demirel, Jeroen Van De Graaf, and Roberto Samarone dos Santos Araújo. Improving Helios with everlasting privacy towards the public. In *EVTW/WTE 2012: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*. USENIX, August 2012. Cited on page 17.

[ECH12] Aleksander Essex, Jeremy Clark, and Urs Hengartner. Cobra: toward concurrent ballot authorization for internet voting. In *EVT/WOTE 2012: Electronic Voting Technology/Workshop on Trustworthy Elections*, page 3. USENIX, 2012. Cited on pages 114 and 115.

[EGHM16] Alex Escala, Sandra Guasch, Javier Herranz, and Paz Morillo. Universal cast-as-intended verifiability. In *FC 2016: 1st Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2016*, pages 233–250. Springer, February 2016. Cited on page 17.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985. Cited on page 9.

[Esp08] Margarita Esponda. Electronic voting on-the-fly with mobile devices. In *ITiCSE 2008: 13th Annual Conference on Innovation and Technology in Computer Science Education*, pages 93–97. ACM, July 2008. Cited on page 38.

[Est10] Estonian National Electoral Committee. E-voting system general overview. http://www.vvk.ee/public/dok/General˙Description˙E-Voting˙2010.pdf, 2010. [Online; accessed 28-March-2017]. Cited on pages 1 and 113.

[FBC+12] Michael Farb, Manish Burman, Gurtej S Chandok, Jon McCune, and Adrian Perrig. Safeslinger: An easy-to-use and secure approach for human trust establishment. Technical report, DTIC Document, 2012. Cited on pages 15 and 14.

[FOO92] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In *AUSCRYPT 1992: 8th International Workshop on the Theory and Application of Cryptographic Techniques*, pages 244–251. Springer, December 1992. Cited on page 113.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO 1986: 5th Annual International Cryptology Conference on Advances in Cryptology*, pages 186–194. Springer, August 1986. Cited on page 10.

[GGP15] David Galindo, Sandra Guasch, and Jordi Puiggali. 2015 neuchâtel's cast-as-intended verification mechanism. In *VoteID 2015: 5th International Conference on E-Voting and Identity*, pages 3–18. Springer, September 2015. Cited on page 127.

[GIR16] Rosario Giustolisi, Vincenzo Iovino, and Peter Rønne. On the possibility of non-interactive e-voting in the public-key setting. In *FC 2016: 1st Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2016*, pages 193–208. Springer, February 2016. Cited on pages 36 and 37.

[GKV+16] Dawid Gawel, Maciej Kosarzecki, Poorvi L. Vora, Hua Wu, and Filip Zagorski. Apollo - end-to-end verifiable internet voting with recovery from vote manipulation. Cryptology ePrint Archive, Report 2016/1037, 2016. `http://eprint.iacr.org/2016/1037`. Cited on pages 2 and 17.

[GM16] Sandra Guasch and Paz Morillo. How to challenge and cast your e-vote. In *FC 2016: 20th international conference on Financial Cryptography and Data Security*. IFCA, February 2016. Cited on pages 17 and 126.

[Gol98] Oded Goldreich. Secure multi-party computation. Manuscript. Preliminary version, 1998. Cited on page 38.

[GRBR13] Gurchetan S Grewal, Mark Dermot Ryan, Sergiu Bursuc, and Peter YA Ryan. Caveat coercitor: Coercion-evidence in electronic voting. In *SP 2013: 34th IEEE Symposium on Security and Privacy*, pages 367–381. IEEE, May 2013. Cited on page 115.

[Gro04] Jens Groth. Efficient maximal privacy in boardroom voting and anonymous broadcast. In *FC 2004: 8th International Conference on Financial Cryptography*, pages 90–104. Springer, February 2004. Cited on pages 36 and 37.

[Gro10] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. *Journal of Cryptology*, 23(4):546–579, 2010. Cited on pages 13 and 14.

[HB16] Thomas Haines and Xavier Boyen. Votor: conceptually simple remote voting against tiny tyrants. In *ACSW 2016: 39th Australasian Computer Science Week Multiconference*, page 32. ACM, February 2016. Cited on page 114.

[HK13] Rolf Haenni and Reto E Koenig. A generic approach to prevent board flooding attacks in coercion-resistant electronic voting schemes. *Computers & Security*, 33:59–69, 2013. Cited on pages 125 and 126.

[HRZ10] Feng Hao, Peter YA Ryan, and Piotr Zielinski. Anonymous voting by two-round public discussion. *IET Information Security*, 4(2):62–67, 2010. Cited on pages 36 and 37.

[HS00] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In *EUROCRYPT 2000: 19th International Conference on the Theory and Applications of Cryptographic Techniques*, pages 539–556. Springer, May 2000. Cited on page 114.

[HS11] Rolf Haenni and Oliver Spycher. Secure internet voting on limited devices with anonymized DSA public keys. In *EVT/WOTE 2001: Electronic Voting Technology/Workshop on Trustworthy Elections*, pages 8–8. USENIX, 2011. Cited on pages 8, 113, and 114.

[HT15] J Alex Halderman and Vanessa Teague. The New South Wales Ivote system: Security failures and verification flaws in a live online election. In *VoteID 2015: 5th International Conference on E-Voting and Identity*, pages 35–53. Springer, September 2015. Cited on page 1.

[HW14] Sven Heiberg and Jan Willemson. Verifiable internet voting in Estonia. In *EVOTE 2014: 6th International Conference on Electronic Voting, Verifying the Vote*, pages 1–8. IEEE, October 2014. Cited on page 127.

[IAC16] IACR. IACR Elections. http://www.iacr.org/elections, 2016. [Online; accessed 19-January-2017]. Cited on pages xi, xiii, 1, 2, 16, 127, vii, and ix.

[JCJ05] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *WPES 2005: 4th ACM workshop on Privacy in the electronic society*, pages 61–70. ACM, November 2005. Cited on pages 8, 99, 114, and 127.

[JJ00] Markus Jakobsson and Ari Juels. Mix and match: Secure function evaluation via ciphertexts. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000: 6th International Conference on the Theory and Application of Cryptology and Information Security*, pages 162–177. Springer, December 2000. Cited on pages 13, 14, and 15.

[JP06] H.L. Jonker and W. Pieters. Receipt-freeness as a special case of anonymity in epistemic logic. In *WOTE 2006: IAVoSS Workshop On Trustworthy Elections*. Robinson College, 2006. Cited on page 116.

[KHF11] Reto Koenig, Rolf Haenni, and Stephan Fischli. Preventing board flooding attacks in coercion-resistant electronic voting schemes. In *IFIP TC 2011: 26th IFIP TC 11 International Information Security Conference*, pages 116–127. Springer, June 2011. Cited on pages 115, 125, and 126.

[KKO+11] Fatih Karayumak, Michaela Kauer, M Maina Olembo, Tobias Volk, and Melanie Volkamer. User study of the improved Helios voting system interfaces. In *STAST 2011: 1st Workshop on Socio-Technical Aspects in Security and Trust*, pages 37–44. IEEE, September 2011. Cited on page 18.

[KMNV16] Oksana Kulyk, Karola Marky, Stephan Neumann, and Melanie Volkamer. Introducing proxy voting to Helios. In *ARES 2016: 11th International Conference on Availability, Reliability and Security*, pages 98–106. IEEE, September 2016. Cited on pages 7 and 41.

[KMNV17] Oksana Kulyk, Karola Marky, Stephan Neumann, and Melanie Volkamer. Enabling vote delegation in boardroom voting [to appear]. In *FC 2017: 2nd Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2017*. IFCA, April 2017. In press. Cited on page 63.

[KNM+16] Oksana Kulyk, Stephan Neumann, Karola Marky, Jurlind Budurushi, and Melanie Volkamer. Coercion-resistant proxy voting. In *IFIP SEC 2016: 31st International Conference on ICT Systems Security and Privacy Protection*, pages 3–16. Springer, June 2016. Cited on page 60.

[KNV+14] Oksana Kulyk, Stephan Neumann, Melanie Volkamer, Christian Feier, and Thorben Koster. Electronic voting with fully distributed trust and maximized flexibility regarding ballot design. In *EVOTE 2014: 6th International Conference on Electronic Voting, Verifying the Vote*, pages 1–10. IEEE, October 2014. Cited on pages 7 and 27.

[KR05] Steve Kremer and Mark Ryan. Analysis of an electronic voting protocol in the applied pi calculus. In *ESOP 2005: 14th European Symposium on Programming, Programming Languages and Systems*, pages 186–200. Springer, April 2005. Cited on pages 8, 115, 116, and 7.

[KRS10] Steve Kremer, Mark Ryan, and Ben Smyth. Election verifiability in electronic voting protocols. In *ESORICS 2010: 15th European Symposium on Research in Computer Security*, pages 389–404. Springer, September 2010. Cited on pages 16 and 115.

[KSRH12] Dalia Khader, Ben Smyth, Peter YA Ryan, and Feng Hao. A fair and robust voting system by broadcast. In *EVOTE 2012: 5th International Conference on Electronic Voting*, volume 205, pages 285–299. Gesellschaft für Informatik, July 2012. Cited on pages 36 and 37.

[KT09] Ralf Küsters and Tomasz Truderung. An epistemic approach to coercion-resistance for electronic voting protocols. In *SP 2009: 30th IEEE Symposium on Security and Privacy*, pages 251–266. IEEE, May 2009. Cited on pages 115 and 116.

[KTV10a] R. Küsters, T. Truderung, and A. Vogt. A game-based definition of coercion-resistance and its applications. In *CSFS 2010: 23rd IEEE Computer Security Foundations Symposium*, pages 122–136. IEEE, July 2010. Cited on pages 98 and 104.

[KTV10b] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: Definition and relationship to verifiability. In *CCS 2010: 17th ACM Conference on Computer and Communications Security*, pages 526–535. ACM, April 2010. Cited on pages 110 and 115.

[KTV12] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Clash attacks on the verifiability of e-voting systems. In *SP 2012: 33rd IEEE Symposium on Security and Privacy*, pages 395–409. IEEE, May 2012. Cited on pages 2 and 16.

[KTV15] Oksana Kulyk, Vanessa Teague, and Melanie Volkamer. Extending Helios towards private eligibility verifiability. In *VoteID 2015: 5th International Conference on E-Voting and Identity*, pages 57–73. Springer, September 2015. Cited on pages 7 and 78.

[KY02] Aggelos Kiayias and Moti Yung. Self-tallying elections and perfect ballot secrecy. In *PKC 2002: 5th International Workshop on Public Key Cryptography*, pages 141–158. Springer, August 2002. Cited on pages 36 and 37.

[KZ07] Mirosław Kutyłowski and Filip Zagórski. Verifiable internet voting solving secure platform problem. In *IWSEC 2007: 2nd International Workshop on Security, Advances in Information and Computer Security*, pages 199–213. Springer, October 2007. Cited on page 115.

[KZZ15] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. End-to-end verifiable elections in the standard model. In *EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Part II*, pages 468–498. Springer, April 2015. Cited on page 115.

[KZZ16] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang. Auditing for privacy in threshold pke e-voting. *International Journal of Information and Computer Security*, 2016. Cited on pages 16, 19, 21, and 23.

[LBD+03] Byoungcheon Lee, Colin Boyd, Ed Dawson, Kwangjo Kim, Jeongmo Yang, and Seungjae Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *ICISC 2003: 6th International Conference on Information Security and Cryptology*, pages 245–258. Springer, November 2003. Cited on page 114.

[LH15] Philipp Locher and Rolf Haenni. Verifiable internet elections with everlasting privacy and minimal trust. In *VoteID 2015: 5th International Conference on E-Voting and Identity*, pages 74–91. Springer, September 2015. Cited on pages 113 and 114.

[LH16] Philipp Locher and Rolf Haenni. Receipt-free remote electronic elections with everlasting privacy. *Annals of Telecommunications*, 71(7):323–336, 2016. Cited on pages 97, 114, and 126.

[LHK16] Philipp Locher, Rolf Haenni, and Reto E Koenig. Coercion-resistant internet voting with everlasting privacy. In *FC 2016: 1st Workshop on Advances in Secure Electronic Voting Associated with Financial Crypto 2016*. Springer, February 2016. Cited on pages 97, 114, and 126.

[LK02] Byoungcheon Lee and Kwangjo Kim. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In *ICISFC 2002: 5th International Conference on Information Security and Cryptology*, pages 389–406. Springer, March 2002. Cited on pages 8 and 9.

[LSBV10] Lucie Langer, Axel Schmidt, Johannes Buchmann, and Melanie Volkamer. A taxonomy refining the security requirements for electronic voting: analyzing Helios as a proof of concept. In *ARES 2010: 7th International Conference on Availability, Reliability and Security*, pages 475–480. IEEE, February 2010. Cited on pages 7, 23, and 32.

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982. Cited on pages 16 and 15.

[M+10] Anastasia Meletiadou et al. *Moderne Instant-Messaging-Systeme als Plattform für sicherheitskritische kollaborative Anwendungen.* PhD thesis, University of Koblenz-Landau, 2010. Cited on page 36.

[Men09] Bo Meng. A critical review of receipt-freeness and coercion-resistance. *Information Technology Journal*, 8(7):934–964, 2009. Cited on page 116.

[MHS12] Murat Moran, James Heather, and Steve Schneider. Verifying anonymity in voting systems using CSP. *Formal Aspects of Computing*, 26(1):63–98, December 2012. Cited on page 115.

[MN06] Tal Moran and Moni Naor. Receipt-free universally-verifiable voting with everlasting privacy. In *CRYPTO 2006: 26th Annual International Cryptology Conference on Advances in Cryptology*, pages 373–392. Springer, 2006. Cited on pages 8, 115, and 116.

[MTM16] Patrick McCorry, Ehsan Toreini, and Maryam Mehrnezhad. Removing trusted tallying authorities. Technical report, Newcastle University, 2016. Cited on page 36.

[Neu16] Stephan Neumann. *Evaluation and Improvement of Internet Voting Schemes Based on Legally-Founded Security Requirements.* PhD thesis, Technische Universität Darmstadt, 2016. Cited on page 7.

[NFVK13] Stephan Neumann, Christian Feier, Melanie Volkamer, and Reto Koenig. Towards a practical jcj / civitas implementation. In *Informatik 2013: 43. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Informatik angepasst an Mensch, Organisation und Umwelt*, pages 804–818, September 2013. Cited on page 115.

[NORV14] Stephan Neumann, M. Maina Olembo, Karen Renaud, and Melanie Volkamer. Helios verification: To alleviate, or to nominate: Is that the question, or shall we have both? In *EGOVIS 2014: 3rd International Conference on Electronic Government and the Information Systems Perspective*, pages 246–260. Springer, September 2014. Cited on page 18.

[NR06] L. H. Nguyen and A. W. Roscoe. Efficient group authentication protocol based on human interaction. In *FCS-ARSPA 2006: Workshop on Foundation of Computer Security and Automated Reasoning Protocol Security Analysis*, pages 9–33, August 2006. Cited on page 14.

[OKNV12] Maina Olembo, Anna Kahlert, Stephan Neumann, and Melanie Volkamer. Partial verifiability in POLYAS for the GI elections. In *EVOTE 2012: 5th International Conference on Electronic Voting*, volume 205, pages 95–109. Gesellschaft für Informatik, July 2012. Cited on page 1.

[Ped91] Torben Pryds Pedersen. A threshold cryptosystem without a trusted party. In *EUROCRYPT 1991: 10th Workshop on the Theory and Application of of Cryptographic Techniques*, pages 522–526. Springer, April 1991. Cited on pages 13 and 12.

[Ped92a] Torben Pryds Pedersen. Distributed provers and verifiable secret sharing based on the discrete logarithm problem. *DAIMI Report Series*, 21(388), 1992. Cited on pages 13 and 12.

[Ped92b] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO 1992: 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 129–140. Springer, August 1992. Cited on page 14.

[Per16] Olivier Pereira. Internet voting with Helios. In *Real-World Electronic Voting: Design, Analysis and Deployment*, pages 277–308. CRC Press, 2016. Cited on pages 20 and 19.

[PR16] Sunoo Park and Ronald L. Rivest. Towards secure quadratic voting. Cryptology ePrint Archive, Report 2016/400, April 2016. `http://eprint.iacr.org/2016/400`. Cited on page 18.

[Pri17] Princeton University. Princeton Undergraduate Elections. https://princeton.heliosvoting.org, 2017. [Online; accessed 28-March-2017]. Cited on pages 1 and 2.

[RBH+09] Peter YA Ryan, David Bismark, James Heather, Steve Schneider, and Zhe Xia. Prêt à voter: a voter-verifiable voting system. *IEEE transactions on*

information forensics and security, 4(4):662–673, 2009. Cited on pages 8, 9, and 7.

[Rit14] Jürg Ritter. Decentralized e-voting on android devices using homomorphic tallying. Master's thesis, Bern University of Applied Sciences, Biel, Switzerland, 2014. Cited on page 37.

[Roe16] Peter Roenne. Jcj with improved verifiability guarantees. In *E-Vote-ID 2016: 1st Joint International Conference on Electronic Voting*. Springer, October 2016. Cited on page 115.

[RRI15] Peter Y A Ryan, Peter B Roenne, and Vincenzo Iovino. Selene: Voting with transparent verifiability and coercion-mitigation. Cryptology ePrint Archive, Report 2015/1105, November 2015. `http://eprint.iacr.org/`. Cited on page 17.

[RW11] Mariana Raykova and David Wagner. Verifable remote voting with large scale coercion resistance. Technical report, Tech. Rep. CUCS-041-11, Columbia, 2011. Cited on page 115.

[SB13] Ben Smyth and David Bernhard. Ballot secrecy and ballot independence coincide. In *ESORICS 2013: 18th European Symposium on Research in Computer Security*, pages 463–480. Springer, September 2013. Cited on pages 11 and 17.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4(3):161–174, August 1991. Cited on pages 10, 11, and 9.

[SCH+14] Sriramkrishnan Srinivasan, Chris Culnane, James Heather, Steve Schneider, and Zhe Xia. Countering ballot stuffing and incorporating eligibility verifiability in Helios. In *NSS 2014: International Conference on Network and System Security*, pages 335–348. Springer, October 2014. Cited on page 17.

[SFC15] Ben Smyth, Steven Frink, and Michael R. Clarkson. Election verifiability: Cryptographic definitions and an analysis of Helios and jcj. Cryptology ePrint Archive, Report 2015/233, March 2015. `http://eprint.iacr.org/`. Cited on pages 7, 8, and 115.

[SFD+14] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the Estonian internet voting system. In *CCS 2014: 21st ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, November 2014. Cited on page 1.

[SGM+15] Uwe Serdult, Micha Germann, Fernando Mendez, Alicia Portenier, and Christoph Wellig. Fifteen years of internet voting in Switzerland [history, governance and use]. In *ICEDEG 2015: 2nd International Conference on eDemocracy & eGovernment*, pages 126–132. IEEE, April 2015. Cited on page 1.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979. Cited on page 12.

[SHKS11] Michael Schläpfer, Rolf Haenni, Reto Koenig, and Oliver Spycher. Efficient vote authorization in coercion-resistant internet voting. In *VoteID 2011: 3rd International Conference on E-Voting and Identity*, volume 7187, pages 71–88. Springer, September 2011. Cited on pages 114 and 115.

[SKHS11] Oliver Spycher, Reto Koenig, Rolf Haenni, and Michael Schläpfer. A new approach towards coercion-resistant remote e-voting in linear time. In *FC 2011: 15th international conference on Financial Cryptography and Data Security*, pages 182–189. Springer, February 2011. Cited on pages 113, 114, 126, and 115.

[SP15] Alan Szepieniec and Bart Preneel. New techniques for electronic voting. *JETS: USENIX Journal of Election Technology and Systems*, 3(2):46–69, 2015. Cited on page 37.

[SVRH11] Guido Schryen, Melanie Volkamer, Sebastian Ries, and Sheikh Mahbub Habib. A formal approach towards measuring trust in distributed systems. In *SAC 2011: 26th ACM Symposium on Applied Computing*, pages 1739–1745. ACM, March 2011. Cited on page 115.

[Tch12] Angel Tchorbadjiiski. Liquid democracy diploma thesis. *RWTH AACHEN University, Germany*, 2012. Cited on pages 59 and 60.

[TJR16] Masoud Tabatabaei, Wojciech Jamroga, and Peter YA Ryan. Expressing receipt-freeness and coercion-resistance in logics of strategic ability: Preliminary attempt. In *AISe 2016: 1st International Workshop on AI for Privacy and Security*, page 1. ACM, August 2016. Cited on page 116.

[TMT+08] Mehdi Talbi, Benjamin Morin, Valérie Viet Triem Tong, Adel Bouhoula, and Mohamed Mejri. Specification of electronic voting protocol properties using adm logic: Foo case study. In *ICICS 2008: 10th International Conference on Information and Communications Security*, pages 403–418. Springer, October 2008. Cited on page 116.

[TPLT13] Georgios Tsoukalas, Kostas Papadimitriou, Panos Louridas, and Panayiotis Tsanakas. From Helios to Zeus. In *EVTW/WTE 2013: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections*. USENIX, August 2013. Cited on page 17.

[TW10] Björn Terelius and Douglas Wikström. Proofs of restricted shuffles. In *AFRICACRYPT 2010: 3rd International Conference on Cryptology in Africa*, pages 100–113. Springer, May 2010. Cited on pages 13, 14, and 116.

[Vol09] Melanie Volkamer. *Evaluation of Electronic Voting. Requirements and Evaluation Procedures to Support Responsible Election Authorities*, volume 30. Springer, 2009. Cited on page 7.

[Wik09] Douglas Wikström. A commitment-consistent proof of a shuffle. In *ACISP 2009: 13th Australasian Conference on Information Security and Privacy*, pages 407–421. Springer, July 2009. Cited on pages 14 and 13.

[Woo14] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, 2014. Cited on page 37.

[WWIH12] Scott Wolchok, Eric Wustrow, Dawn Isabel, and J Alex Halderman. Attacking the Washington, DC internet voting system. In *FC 2012: 16th International Conference on Financial Cryptography and Data Security*, pages 114–128. Springer, March 2012. Cited on page 1.

[ZHT13] Bernd Zwattendorfer, Christoph Hillebold, and Peter Teufl. Secure and privacy-preserving proxy voting system. In *ICEBE 2013: IEEE 10th International Conference on e-Business Engineering*, pages 472–477. IEEE, September 2013. Cited on pages 59 and 60.

[Zim95] Philip R Zimmermann. PGPfone: Pretty Good Privacy Phone Owner's Manual. MIT, http://web.mit.edu/network/pgpfone/manual, 1995. [Online; accessed 28-March-2017]. Cited on page 15.