Ceremony Analysis Meets Verifiable Voting: Individual Verifiability in Helios

Taciane Martimiano*, Eduardo dos Santos, †, Maina Olembo‡, Jean Everson Martina*,
Ricardo Alexandre Reinaldo de Moraes,*
*Universidade Federal de Santa Catarina
Florianopolis - SC - Brazil
Email: taciane.m@inf.ufsc.br, jean.martina@ufsc.br, ricardo.moraes@ufsc.br
†University of Oxford - Oxford - United Kingdom
Email: eduardo.dossantos@cybersecurity.ox.ac.uk
‡Technische Universitt Darmstadt - Darmstadt - Germany
Email: maina.olembo@cased.de

***Abstract*—The Helios verifiable voting system offers voters an opportunity to verify the integrity of their individual vote, that it is cast, and included in the final count, as intended. While not all voters have to verify, these steps can be cumbersome for those who aim to carry them out. Therefore, new verification processes have been proposed in order to improve usability. Voters can use a web-based verifier provided by any one of several independent verification institutes, or a smartphone app developed and provided by these institutes. In this work, we describe these verification processes as ceremonies, and thus model the human peer's interaction. We undertake a security analysis applying an adaptive threat model suited to analysing human-device and human-human channels. More realistic threats on these channels are identified, compared to those from an analysis using a Dolev-Yao attacker.**

Keywords–Voting; Threat models; Security Ceremonies.

I. INTRODUCTION

In order to engender voter trust in electronic voting, cryptographic voting systems that offer verifiability while maintaining vote secrecy have been proposed, and continue to gain ground. The Scantegrity II end to end verifiable voting system was used in a governmental election [1] and a modified version of the Prêt à Voter system was used in the 2014 Victoria State elections in Australia [2]. In this space, Helios [3][4], an open-source, verifiable, Internet-based voting system, stands out for its continued use, primarily in academic contexts, for example, in 2009, to elect the university president at the Université Catholique de Louvain [4]. It was also used in the 2013 Princeton University undergraduate student government elections [5], and to elect the Board of Directors of the International Association for Cryptologic Research (IACR) [6].

With the use of Helios, it is assumed that voters can and will verify their votes to ensure vote integrity [3]. While it is not known whether this assumption is true for the elections where Helios has been used, findings from expert reviews [7] and user studies [8] suggest that this is not likely to be the case, due to the cumbersome nature of the verification process. Usability improvements to the Helios voting interfaces, with a specific focus on the verification aspect, have been proposed to ensure that this assumption can be met. These improvements involve the voter using verifiers provided by trusted institutes. These verifiers are available in two forms: either accessible to the voter through the institutes' web page, or via download and installation on the voter's smartphone as an app. In this work, we analyse the security implications of these improvements, a practice recommended for usable security [9]. The focus is on verifiability and integrity. We apply ceremony analysis [10] using the adaptive threat model provided by Carlos *et al.* [11], which is appropriate to analyse the human-device and human-human channels. Following standard practice, a Dolev-Yao adversary [12] is assumed on the device-device channels.

A. Contribution

Our findings show that:

1) No threats to secrecy are present when the voter uses the smartphone app to verify;
2) Reputation attacks might be carried out to undermine the institutes participating in elections. In such cases, voter education on necessary steps is required;
3) Semi-formal verification can be applied to election ceremonies.

We discuss the implications of these findings for voting and verification in Helios.

B. Related work

Several extensions to the Helios voting protocol have been proposed, focusing on providing everlasting privacy [13], privacy and correctness [14], and preventing attacks against privacy [15]. Zeus [16] is a verifiable voting and counting system developed based on the Helios version in [3]. The authors propose that the voter enters an *audit code* to indicate that a submitted vote should be verified. However, no analysis of the security implications of these modifications is provided. A variant of Helios that prevents ballot box stuffing is proposed in [14]. Comparatively, the research we report in this paper analyses the security of two proposals made to improve the usability of verification in Helios, in order to ensure that voters can indeed verify that their ballots are cast, and counted in the final tally, as intended.

Carlos *et al.* [11] proposed an adaptive threat model and applied it to analyse the Bluetooth pairing protocol. In our work, we apply their model to a new domain - verifiable voting.

This paper is structured as follows: Section II brings the necessary background on the Helios protocol, analysis on security ceremonies and the adaptive threat model used in this work. Section III shows the ceremony for the institutes' website proposal and its analysis. Section IV presents the ceremony for the app proposal and its analysis. Finally, Section V has our final remarks and conclusions.

 ISBN: 978-1-61208-427-5

II. BACKGROUND

In this section, we first describe the Helios voting protocol and then we provide background information on the design and analysis of ceremonies. Moreover, we introduce the adaptive threat model and the security criteria applied in this work.

A. The Helios voting protocol

We describe the Helios voting protocol, focusing only on those aspects that are relevant to verifiability and that are necessary to understand the proposals presented in later sections.

In order to vote using Helios, the voter downloads the Helios *ballot preparation system* (BPS)[17] onto his web browser. He indicates the candidate(s) of his choice on the ballot. The BPS encrypts these choices (i.e. the vote) and commits to the encryption by displaying a hash value, which we refer to as a check-code. The voter should record the check-code displayed if he plans to verify. At this point, the voter makes a choice to either submit this encrypted vote to the public web bulletin board or to challenge the voting system, verifying whether the vote has been correctly encrypted.

If the voter decides to verify, he interacts with the Helios *ballot verifier system* (BVS). The BPS displays the candidate(s) and the randomness used for encryption. The voter selects and copies this information to the voting device's clipboard and pastes it into the BVS, which BPS opens in a new web browser window. The BVS encrypts the corresponding candidate and generates the hash value of this encryption. This hash value is displayed together with the candidate(s) contained in the vote received earlier. In order to complete the verification process, the voter needs to confirm that the check-code displayed by the BVS matches the one displayed earlier by the BPS. Additionally, he needs to confirm that the vote is correct. If both these conditions are met, the voter is assured that the system correctly encrypted the vote in this instance. He can repeat the verification process several times until he is satisfied that the system is behaving correctly. Once votes are verified they can no longer be submitted to the public web bulletin board as the voter could easily prove how he voted using the revealed randomness. Thus, new randomness is required. As the BVS learns the content of the encryption, the use of test votes that differ from the final vote has been recommended [8], to avoid the BVS computing intermediate results.

If the voter chooses to submit his vote to the public web bulletin board, he is prompted to authenticate himself, and his encrypted vote is then posted on to the public web bulletin board together with the check-code. To verify that the encrypted vote is correctly stored on the voting server or public web bulletin board, the voter needs to confirm that the check-code appears on the public web bulletin board next to his name [3], or some pseudonym [4]. It is only necessary to do this once as the Helios threat model assumes that auditors continuously observe the bulletin board preventing malicious behaviour.

B. The design and analysis of ceremonies

Ceremonies extend protocols by including human peers and allowing the detection of otherwise undetectable security flaws [10]. In protocols, all the human actions are modelled as assumptions and, when the protocol is implemented, these assumptions can result in user interactions that are unrealistic or not well-specified. In ceremonies, additional channels are available to model the interaction of human peers to other peers in the system, namely, the human-human (HH) channel and human-device (HD) channel, besides the device-device (DD) channel from the protocol structure.

To take these additional channels into account, we analyse ceremonies using the adaptive threat model proposed by Carlos *et al.* [11]. This adaptive threat model uses the Dolev-Yao (DY) attacker's set of capabilities, by dynamically adding or removing capabilities from the whole set. Doing so helps in identifying cases where overly stringent requirements are placed on users. While such requirements are motivated by security concerns, they are likely to negatively impact usability. Understanding the correct threat model the user is subject to, when interacting in a ceremony, will prevent him from being overloaded with unrealistic scenarios and guarantee that important security properties will hold [11].

The analysis process begins with the establishment of channels present at the ceremony. This involves listing the human nodes and devices involved, identifying which of these nodes exchange messages and which type of communication channel they represent (i.e., HH, HD or DD). Thus, it is possible to analyse the impact of an attacker's capabilities in each channel. The attacker's goal is to learn the contents being exchanged among nodes. The DY threat model defines abilities that allow the attacker to achieve his goal. Therefore, we observe which approaches the attacker can use to stop or modify messages, create and send messages of their own knowledge, etc. in order to obtain a realistic threat model that includes the profiles of the attackers associated to each channel. For example, if the attacker has access to a given cryptographic key and intercepts messages encrypted with that key, he will be able to decrypt and learn the contents of these messages, compromising the safety of the messages shared by that channel. Interestingly, following the adaptive threat model of Carlos *et al.*, we have a realistic and specific threat model to each ceremony, given its participants, channels and environments to which it will be subject to.

C. An adaptive threat model

Dolev and Yao [12] formalised the attacker model introduced by Needham and Schroeder [18], giving the attacker absolute control of the network, such that the attacker can copy, replay, alter and create messages. However, he cannot perform cryptanalysis. Based on [11], the Dolev-Yao (DY) attacker has the following set of capabilities: Eavesdrop, Initiate, Atomic Break Down, Block, Crypto, Fabricate, Spoof, Reorder, Modify and Replay [11].

Assumptions: We present assumptions of the original Helios system, as well as our assumptions regarding the entities involved in the ceremony, and the ceremony itself.

The *attacker* lies only on the channel as is the usual Dolev-Yao assumption. Further, he cannot control more than one device-device channel.

Any participating *verification institute* is trustworthy as any malicious behaviour would lead to a loss of reputation. We are not considering denial of service attacks.

The *voter* is an honest peer in the ceremony as a dishonest voter can easily prevent a ceremony from concluding correctly.

 ISBN: 978-1-61208-427-5

We also do not consider coercion. Thus, the cases where the attacker is the voter are not included.

Finally, the *ceremony* is assumed to have only one entry and one exit point, so the voter is expected to follow all the steps provided in the ceremony he is executing.

We recognise some of the assumptions are weak. However, our main concern for this work is to establish the simplest version of the presented scenarios. Analysing more complex variations are left for future work.

1) Security criteria: As the adaptive threat model will be applied to the electronic voting context, we define necessary security criteria adapted from Neumann *et al.* [19].

A number of security properties are considered important in the electronic voting context. In this work, we concentrate on verifiability and integrity, likely the most important properties for elections conducted over a remote channel.

Integrity: The sum of all participating voters' submitted votes (votes submitted to and stored on the voting server or the public web bulletin board) matches the declared election result.

Integrity violations must not go undetected [20]. From this requirement, we obtain the definition of verifiability.

Verifiability: Property in which the voter assures himself of the integrity of the individual vote and the public is assured of the integrity of the election result. Verifiability consists of evidence of the following aspects being provided:

- The vote correctly represents the voter's choice;
- The vote has been stored on the voting server or public web bulletin board as it was cast by the voter;
- All valid votes on the public web bulletin board are tallied without modification.

III. Using a web-based ballot verifier

We summarise the processes that voters would carry out using a web-based ballot verifier provided by the trusted institutes. We analyse these processes using the adaptive threat model and briefly compare our results to those obtained in the case of a DY attacker, and close this section with a discussion of the results.

A. Proposal

The message flow for this proposal is seen in Figure 1. The text below the arrows identifies the channels under consideration. V refers to Voter, B to Booth, I to Institute, A to App and BB to Bulletin Board.

The voting process is similar to that described in subsection II-A. Note that differences in the voting and verification processes are reported in [7]. A relevant difference for this ceremony is that the voter enters the URL into the address bar in order to open the election website. He receives the voting credentials via postal mail, rather than clicking on a URL in the invocation email as in the original Helios.

We therefore concentrate on the verification processes where differences emerge between the original Helios and across the proposals presented in this work. In order to verify that his vote is correctly encrypted on the voting device, a voter first needs to record the check-code displayed by the BPS (see message 9 in Figure 1).

The voter then expresses to BPS his intention to verify, in message 10, views the verification institutes that are available in message 11, then selects an institute that he trusts, in message 12. He is re-directed to the selected institute's verification web page, in a new browser window. The BPS transmits the information necessary for verification, that is Vote + R and accompanying proofs to the selected institute, in message 13. Since the vote is transmitted to the institute, the case considered here is that the voter verifies a 'test vote', that is, one that is not equivalent to the final vote that he will cast.

1.	V	$\xrightarrow[HD_1]{}$	B	:	URL
2.	B	$\xrightarrow[HD_1]{}$	V	:	Instructions Page
3.	V	$\xrightarrow[HD_1]{}$	B	:	"*Proceed to ballot*"
4.	B	$\xrightarrow[HD_1]{}$	V	:	Ballot Page
5.	V	$\xrightarrow[HD_1]{}$	B	:	Vote
6.	V	$\xrightarrow[HD_1]{}$	B	:	"*Review vote*"
7.	B	$\xrightarrow[HD_1]{}$	V	:	Review Page
8.	V	$\xrightarrow[HD_1]{}$	B	:	"*Encrypt vote*"
9.	B	$\xrightarrow[HD_1]{}$	V	:	$H\{\{Vote + R\}_{Epk}\}$
10.	V	$\xrightarrow[HD_1]{}$	B	:	"*Check encrypted vote*"
11.	B	$\xrightarrow[HD_1]{}$	V	:	Verifying Options Page
12.	V	$\xrightarrow[HD_1]{}$	B	:	I
13.	B	$\xrightarrow[DD_1]{}$	I	:	Vote + R
14.	I	$\xrightarrow[HD_2]{}$	V	:	$H\{\{Vote + R\}_{Epk}\}$ + Vote

Figure 1. Verifying vote using institutes' website

1.	V	$\xrightarrow[HD_1]{}$	B	:	URL
2.	B	$\xrightarrow[HD_1]{}$	V	:	Instructions Page
3.	V	$\xrightarrow[HD_1]{}$	B	:	"*Proceed to ballot*"
4.	B	$\xrightarrow[HD_1]{}$	V	:	Ballot Page
5.	V	$\xrightarrow[HD_1]{}$	B	:	Vote
6.	V	$\xrightarrow[HD_1]{}$	B	:	"*Review vote*"
7.	B	$\xrightarrow[HD_1]{}$	V	:	Review Page
8.	V	$\xrightarrow[HD_1]{}$	B	:	"*Encrypt vote*"
9.	B	$\xrightarrow[HD_1]{}$	V	:	$H\{\{Vote + R\}_{Epk}\}$
10.	V	$\xrightarrow[HD_1]{}$	B	:	"*Log in to cast vote*"
11.	B	$\xrightarrow[HD_1]{}$	V	:	Login Page
12.	V	$\xrightarrow[HD_1]{}$	B	:	VoterID + Password
13.	B	$\xrightarrow[DD_3]{}$	BB	:	$\{Vote\}_{Epk}$
14.	B	$\xrightarrow[HD_1]{}$	V	:	Vote is submitted
15.	V	$\xrightarrow[HD_1]{}$	B	:	I
16.	B	$\xrightarrow[DD_1]{}$	I	:	$H\{\{Vote + R\}_{Epk}\}$
17.	I	$\xrightarrow[HD_2]{}$	V	:	$H\{\{Vote + R\}_{Epk}\}$ + VoterID

Figure 2. Submitting final vote using institutes' website

The institute will compute the check-code using the information it receives from the booth, and displays this, along with the vote it received, to the voter (message 14). The voter now needs to confirm that the two check-codes match (H_{BPS} = H_{BVS_I}), and that the vote displayed by the institute matches his selection on the ballot.

 ISBN: 978-1-61208-427-5

Next, we describe the process for the voter to verify that his vote is correctly stored on the voting server or public web bulletin board. This message flow is shown in Figure 2. The voter records the check-code displayed to him, in message. He then logs in to submit his vote, in messages 10 - 12. Upon successful authentication, the booth submits the vote to the bulletin board, in message 13. The voter would select an institute from several options displayed after he submits his vote, in message 15. A new web page would open. He would enter the recorded check-code information and view the result returned by the institute, in message 17.

Note that the institute additionally needs to display the Voter ID in its response to the voter, in order to prevent a successful clash attack [21], where different voters are shown the same check-code when they verify that their vote has been stored on the bulletin board. This can only be detected if the Voter ID is returned as it is a unique value.

B. Analysis

We apply the adaptive threat model in our ceremonies so we can conclude which scenarios are realistic and whether the attacker succeeds or not in his attempts to corrupt the system. For more specific and detailed information about the notation, formulae and semantics of the proofs, see Carlos et al[11].

1) Preliminaries: With the full Dolev-Yao (DY) attacker capabilities in mind and applying the framework proposed by Carlos *et al.* [11], we can evaluate our ceremonies against a less powerful and more realistic variation of such an attacker. Thus, we analyse the threat model each of the communication channels is subject to in each of the scenarios studied. We describe the adaptive threat model for each of the ceremonies and compare the results to the DY threat model. In the latter case, all the communication channels are under a full DY attacker.

In the adaptive threat model, only the device-device (DD) channel is under a full DY attacker, while the human-device (HD) channel is under a DY-E attacker. DY-E means that the attacker has all the DY capabilities, but lacks the eavesdrop capability. This capability is excluded since we are considering controlled environments, where the voter does not need to check around if there is someone eavesdropping his actions.

Considering the HD channel, we assume there is a human being (and not a machine pretending to be a human peer) communicating with a device. Thus, the voter interacts with his device (for example, looking at his computer screen or typing in the keyboard). A DY - E attacker is not able to compromise the secrecy of the messages sent through HD channels. This assumption is justified because the voter has control of his computer, thus limiting the attacker's actions. Therefore, once the attacker is not able to learn any voter's information, he can only apply his other capabilities over the knowledge he already has. For example, even if the attacker fabricates messages or uses his crypto capability, he can only use his own knowledge, which poses no threat to the voting and verifying ceremonies. We show proofs informing the knowledge each peer of the system and the attacker have through the set knows(Y), representing the set of knowledge of an agent Y in the ceremony [11].

We now move to the actual analysis of the ceremonies involving the institutes.

2) Test vote is correctly encrypted on the voting device: Based on Figure 1:

If the messages M1 to M12, and message M14 are run against a DY-E attacker, and message M13 is run against a DY attacker, the attacker (Att) can prevent the institute I from learning Vote + R and instead send $Vote_{att}$ + R instead, where $Vote_{att}$ is chosen by the attacker.

$$\frac{(M_{1...12,14} \cup DY - E) \wedge (M_{13} \cup DY)}{\begin{array}{r} Vote \wedge R \wedge Vote_{att} \in knows(Att) \wedge \\ Vote \in knows(B) \wedge \\ (Vote + R) \notin knows(I) \wedge (Vote_{att} + R) \in knows(I) \end{array}}$$

a) Proof: Assume the attacker *Att* initiated two simultaneous pairing sessions between the booth B and the institute I in message 13. *Att* uses his block, atomic breakdown, fabricate and initiate capabilities in this message, preventing I from learning the correct vote and randomness information, that is (Vote+R), forcing it to receive ($Vote_{att}$ +R) instead.

3) Final vote is correctly stored on the voting server or public web bulletin board: Based on Figure 2:

If the messages M1 to M12, M14, M15 and M17 are run against a DY-E attacker, and message M13 and M16 are run against a DY attacker, the attacker (Att) can prevent the bulletin board BB from receiving the correct $\{Vote\}_{Epk}$. Att can also prevent the institute I from learning $H\{\{Vote + R\}_{Epk}\}$. Instead, he sends altered information $\{Vote_{att}\}_{Epk}$ and $H_{att}\{\{Vote_{att} + R_{att}\}_{Epk}\}$ to the bulletin board BB and the institute I, respectively, where $Vote_{att}$ is chosen by the attacker. Then, the attacker uses his crypto capability to generate the $\{Vote_{att}\}_{Epk}$ information and his fabricate capability to generate $H_{att}\{\{Vote_{att}\} + R_{att}\}_{Epk}\}$.

$$\frac{(M_{1...12,14,15,17} \cup DY - E) \wedge (M_{13,16} \cup DY)}{\begin{array}{r} \{Vote\}_{Epk} \wedge \{Vote_{att}\}_{Epk} \wedge H\{\{Vote + R\}_{Epk}\} \wedge \\ H_{att}\{\{Vote_{att} + R_{att}\}_{Epk}\} \in knows(Att) \wedge \\ Vote \in knows(B) \wedge \{Vote\}_{Epk} \notin knows(BB) \wedge \\ \{Vote_{att}\}_{Epk} \in knows(BB) \wedge \\ H\{\{Vote + R\}_{Epk}\} \notin knows(I) \wedge \\ H_{att}\{\{Vote_{att} + R_{att}\}_{Epk}\} \in knows(I) \end{array}}$$

a) Proof: We assume the attacker *Att* initiated two simultaneous pairing sessions between booth B and the public web bulletin board BB maintained by institute I. The attacker Att uses his block, fabricate and initiate capabilities (message 13 in Figure 2) and sends to the bulletin board BB $\{Vote_{att}\}_{Epk}$, instead.

Note that the attacker can know the existing votes and the public key of the election, however the attacker cannot know the randomness information R.

C. Results

If we consider scenarios with the DY threat model, the attacker has total control of all channels and is able to manipulate the voter in the whole voting process. In these scenarios, the attacker intercepts all messages, sending messages in his knowledge to booth B, institute I, and bulletin board BB,

instead of the original messages. At the same time, he displays to the voter the right content, pretending to be the legitimate entities. Therefore, the voter is led to believe that his vote was encrypted, submitted and stored as intended when this is not the case. Nevertheless, such a scenario is highly unlikely to happen in real world situations, as the HD channel limits the attacker's actions. Furthermore, by involving the human peer, it is difficult for the attacker to control this channel and the information being exchanged without being noticed.

A scenario that is realistic and feasible, involves the attacker intercepting messages on the DD channel. Therefore, the institute receives altered information and calculates a different check-code from the one expected by the voter. The voter then no longer trusts the institute, believing it to be unreliable. This result highlights the need for multiple institutes to be available, providing verification services to voters. The voter is free to verify with several other institutes. If these subsequent checks also return a failed result, he can then contact the election commission.

Analysing the two ceremonies presented above, we can see Vote+R being sent without any encryption in the first ceremony while the second one contains the vote encrypted with the public key of the election (E_{pk}). From this we can conclude that secrecy does not hold in the ceremony for the test vote represented in Figure 1. Secrecy does hold when the voter decides to cast his final vote, represented in Figure 2.

We can conclude this given the fact that even when the DY attacker intercepts message 13 in the DD channel, he only sees a check-code that does not give any information about the vote or the randomness information used. The attacker thus cannot know the vote.

IV. Using a smartphone app and QR codes

We describe the processes that voters would carry out to verify with a verifier installed as an app on the smartphone. With this proposal, the voter has a way to verify that is separate from the voting device. Thus, there is no longer a need to trust the voting device with respect to integrity. Additionally, it uses a device that is in the voter's possession and that he likely trusts (with respect to secrecy and integrity). We analyse these ceremonies using an adaptive threat model, and compare our findings to those using a DY attacker. We conclude this section with the results of our analysis.

A. Proposal

The message flow for this proposal is shown in Figure 3. The BPS displays a QR code containing the check-code in addition to the human-readable value, in message 9. The voter opens the smartphone app and scans the QR code containing the check-code, in messages 10 and 11. This check-code will be stored by the app for later use during the verification process. The voter then expresses his intention to verify to the voting booth, in message 12.

He scans a second QR code, in message 15, and the app computes the check-code comparing it to the first check-code. It then informs the voter that the check-codes match (in a success scenario), and prompts him to confirm that the displayed vote matches his initial input on the ballot, in messages 16 - 17. This is an implementation of a *forcing function* [22] preventing the voter from proceeding without confirming that his vote is correct.

1.	V	$\xrightarrow[HD_1]{}$	B	:	URL
2.	B	$\xrightarrow[HD_1]{}$	V	:	Instructions Page
3.	V	$\xrightarrow[HD_1]{}$	B	:	"*Proceed to ballot*"
4.	B	$\xrightarrow[HD_1]{}$	V	:	Ballot Page
5.	V	$\xrightarrow[HD_1]{}$	B	:	Vote
6.	V	$\xrightarrow[HD_1]{}$	B	:	"*Review vote*"
7.	B	$\xrightarrow[HD_1]{}$	V	:	Review Page
8.	V	$\xrightarrow[HD_1]{}$	B	:	"*Encrypt vote*"
9.	B	$\xrightarrow[HD_1]{}$	V	:	Check-code + $QRCode^*$
10.	V	$\xrightarrow[HD_3]{}$	A	:	"*Open/Scan*"
11.	B	$\xrightarrow[DD_2]{}$	A	:	QRCode
12.	V	$\xrightarrow[HD_1]{}$	B	:	"*Check encrypted vote with app*"
13.	B	$\xrightarrow[HD_1]{}$	V	:	QRCode Page
14.	V	$\xrightarrow[HD_3]{}$	A	:	"*Check that vote is correctly encrypted*"
15.	B	$\xrightarrow[DD_2]{}$	A	:	$QRCode^{**}$
16.	A	$\xrightarrow[HD_3]{}$	V	:	"*Check-codes match. Is this your vote?*" + Vote
17.	V	$\xrightarrow[HD_3]{}$	A	:	Confirm (Yes)
18.	V	$\xrightarrow[HD_3]{}$	A	:	"*Close*"

*Check-code is the hash $H\{\{Vote + R\}_{Epk}\}$. In messages 9 and 11, the QRCode has the check-code contents.

**In messages 13 and 15, the QRCode has $(Vote + R + E_{pk})$ information.

Figure 3. Verifying vote using institutes' app

1.	V	$\xrightarrow[HD_1]{}$	B	:	URL
2.	B	$\xrightarrow[HD_1]{}$	V	:	Instructions Page
3.	V	$\xrightarrow[HD_1]{}$	B	:	"*Proceed to ballot*"
4.	B	$\xrightarrow[HD_1]{}$	V	:	Ballot Page
5.	V	$\xrightarrow[HD_1]{}$	B	:	Vote
6.	V	$\xrightarrow[HD_1]{}$	B	:	"*Review vote*"
7.	B	$\xrightarrow[HD_1]{}$	V	:	Review Page
8.	V	$\xrightarrow[HD_1]{}$	B	:	"*Encrypt vote*"
9.	B	$\xrightarrow[HD_1]{}$	V	:	Check-code + $QRCode^*$
10.	V	$\xrightarrow[HD_3]{}$	A	:	"*Open/Scan*"
11.	B	$\xrightarrow[DD_2]{}$	A	:	QRCode
12.	V	$\xrightarrow[HD_1]{}$	B	:	"*Log in to cast vote*"
13.	B	$\xrightarrow[HD_1]{}$	V	:	Login Page
14.	V	$\xrightarrow[HD_1]{}$	B	:	VoterID + Password
15.	B	$\xrightarrow[DD_3]{}$	BB	:	$\{Vote\}_{Epk}$
16.	B	$\xrightarrow[HD_1]{}$	V	:	Vote is submitted
17.	V	$\xrightarrow[HD_3]{}$	A	:	"*Check that vote is on bulletin board*"
18.	A	$\xrightarrow[DD_4]{}$	BB	:	$H\{\{Vote + R\}_{Epk}\}$
19.	BB	$\xrightarrow[DD_4]{}$	A	:	Confirmation + VoterID

*Check-code is the hash $H\{\{Vote + R\}_{Epk}\}$. In messages 9 and 11, the QRCode has the check-code contents.

Figure 4. Submitting final vote using institutes' app

In order to verify that a vote is correctly stored on the voting server or public web bulletin board (see Figure 4), the voter scans the first QR code containing the check-code as in the previous ceremony, in message 10. He logs in (messages 12 - 14) and the booth submits his vote upon successful authentication, in message 15. The voter instructs the app to check the public web bulletin board for the stored check-code, in message 17. The app does this by querying the public web bulletin board for the check-code, in message 18. In a success scenario, it displays a message to the voter that the check-code

was stored on the public web bulletin board. To prevent clash attacks [21], the app should return the accompanying voter ID as well. In a failure scenario, the voter will be informed that the check-code was not found. He can use other apps for verification. In order for the ceremony to terminate, the voter will be directed to contact the election commission should multiple checks with various apps return failed results.

B. Analysis

Before we present the analysis and results, we first review necessary considerations for the adaptive threat model.

1) Preliminaries: Although DD channels are usually under the full DY attacker, it is not realistic in the DD_2 channel (for example, message 11 in Figure 3). This channel characterises a 'visual channel' once there is no Bluetooth or connection of any kind between these devices. We are considering the ideal case where the smartphone has no virus or worms, for simplicity. Nonetheless, instructions regarding safety or the lack thereof could be given to the user, so he would be aware of the threat model he is subject to and decide whether continue or quit the remaining proceedings.

Channel DD_2 represents the scenario where the voter is using his smartphone to scan the QR code displayed by the computer. Both devices are considered to be in voter's possession, and not under the attacker's control. We have a similar situation as described for the HD channel (in Section III-B1). For example, it is clear that the attacker cannot block any contents passing through the DD_2 channel as this would imply the attacker blocking the computer screen from the voter and his smartphone. The same holds if the attacker tries to perform any of his other capabilities. Thus, for a successful attack, the attacker has to possess the voter's devices. This is only feasible if the voter leaves the devices unattended in the middle of the vote casting process.

Therefore, we consider the DD_2 channel as being DY - E, similar to the HD channel. Any weakened variation of the DY - E attacker (any combination of the capabilities of the full DY attacker, less eavesdrop) can be used because this attacker will not be effective. This is due to the fact that eavesdrop is the only capability which can compromise the secrecy of the voter's vote.

We now move to analyse each ceremony involving the app individually, using the adaptive threat model, and the DY threat model, respectively.

2) Verify vote is correctly encrypted on the voting device:: Based on Figure 3:

If all messages M1 to M18 are run against a DY-E attacker, the attacker cannot perform any significant attack with respect to secrecy and integrity.

$$\frac{M_{1...18} \cup DY - E}{\emptyset}$$

a) Proof: For this ceremony, we consider that the DD_2 channel (in messages 11 and 15 of Figure 3) is not under a full DY attacker. This means that these channels are a weakened variation of the DY threat mode,l because the scenario involves a computer displaying a QR code and communicating with the voter's mobile device. Thus, once the attacker is unable to eavesdrop on the communication, all the other capabilities the attacker has will not have an effect on the secrecy of the voter's vote in the ceremony. Therefore, as the attacker cannot possess the voter's devices and he cannot know the voter's vote, he can no longer perform any significant attack.

3) Final vote is correctly stored on the voting server or public web bulletin board: Based on Figure 4:

Consider messages M1 to M14, M16 and M17 are run against a DY-E attacker, and messages M15, M18 and M19 are run against a DY attacker. The attacker (Att) *can prevent the bulletin board BB from learning the correct values of* $\{Vote\}_{Epk}$ *and* $H\{\{Vote + R\}_{Epk}\}$.

$$\frac{(M_{1...14,16,17} \cup DY - E) \wedge (M_{15,18,19} \cup DY)}{\begin{array}{r} Vote \wedge Vote_{att} \wedge \{\text{Vote}\}_{Epk} \wedge \\ \{\text{Vote}_{att}\}_{Epk} \wedge \text{H}\{\{\text{Vote + R}\}_{Epk}\} \wedge \\ \text{H}_{att}\{\{Vote_{att} + R_{att}\}_{Epk}\} \in knows(Att) \wedge \\ Vote \in knows(B) \wedge \\ \{\text{Vote}\}_{Epk} \wedge \text{H}\ \{\{\text{Vote + R}\}_{Epk}\} \notin knows(BB) \wedge \\ \{\text{Vote}_{att}\}_{Epk} \wedge \text{H}_{att}\{\{Vote_{att} + R_{att}\}_{Epk}\} \in knows(BB) \end{array}}$$

a) Proof: We assume the attacker *Att* initiated two simultaneous pairing sessions between the booth B and the bulletin board BB (message 15 in Figure 4) and between the app A and BB (messages 18 and 19 in Figure 4). We continue to assume that the DD_2 channel has a DY-E attacker since it is a visual channel. The attacker *Att* uses his capabilities of block, fabricate and initiate in messages 15, 18 and 19 where *Att* sends to the bulletin board BB a different value of the encrypted vote and a different value of the check-code, instead of the original ones.

C. Results

If we consider the DY threat model, the attacker can manipulate the voter by manipulating the information displayed to him. Such a situation can be considered realistic (see Figure 4). However, it is highly unlikely to happen due to the fact that HD channels are secure under our assumption that the environment is controlled. In addition, we demonstrated it is unrealistic (Figure 3). This ceremony is more secure due to the presence of the visual channel, which limits the attacker's actions, once it has the same behaviour as on the HD channels. A very important contribution of the app proposal is that the test and final votes are both secret, when compared to the proposal that uses the institutes (where the test vote is sent in clear through a full DY channel). Such a contribution means that the security property of secrecy holds in the app ceremony and, as the messages are no longer interrupted and modified, we can conclude this ceremony also ensures integrity.

In the ceremonies involving verifying using the app, the attacker cannot control more than one DD channel [11]. Therefore, either the attacker controls the message between the booth B and the bulletin board BB (message 15 of Figure 4) or he controls the messages between the app A and the bulletin board BB (message 18 of Figure 4). When the attacker succeeds, the bulletin board BB does not display to the voter the expected confirmation (message 19 of Figure 4). In such a situation, the voter would be advised to contact the election commission.

 ISBN: 978-1-61208-427-5

V. CONCLUSION

Two verification proposals have been made to improve the usability of the individual verifiability processes in the Helios voting system. We have analysed the security of these proposals using a framework proposed by Carlos *et al.* [11], which uses an adaptive threat model. To this end, we considered the voting and verification processes in Helios as ceremonies in order to include the human peer's interaction in our analysis. The Dolev-Yao adversary model is shown to bestow unrealistic powers on the attacker. Hence, an adaptive threat model is applicable in analysing the voting and verifying ceremonies.

The proofs presented in this paper were subject to formal verification using the theorem prover SPASS and the same technique by Carlos et. al. Again due to space constraints they were not included but are available at:
https://github.com/tacianem/HeliosSpass

The first verification proposal involves the voter verifying using a web-based verifier provided by a trusted institute. Our results show the possibility of secrecy violations when the voter verifies that his vote is correctly encrypted on the voting device, and integrity violations when he also verifies that his vote is correctly submitted to the bulletin board. The secrecy violations would only arise if voters did not verify test votes. Integrity violations, on the other hand, would take the form of 'reputation' attacks, resulting in the voter mistrusting the institute as he detects that it displays incorrect information.

The situation is seen to improve with regard to the smartphone app. First, secrecy is maintained due to the presence of a visual channel, and because information is transmitted in encrypted form. Our results also show that no significant attack can occur when the voter verifies that his vote is correctly encrypted on the voting device. Integrity violations are as in the previous case, where reputation attacks would lead to a mistrust of the participating institute.

One can argue that the impact of the reputation attacks is low due to the availability of mitigating strategies. We highlight that integrity assurances in both verification proposals rest on the distribution of trust, that is, there are several options, whether web-based verifiers, or smartphone apps from trusted institutes, available for the voter to use. Should the verification process fail in one case, the voter can verify using other sources. Indeed, voters who do not want to trust any of the available institutes can use all the provided verification mechanisms. We acknowledge that this is not an ideal case with regard to usability, however, we note that it places less of a burden on the voter than would be the case if a more powerful adversary was considered.

The results of this work have highlighted several improvements that can be made to the Helios voting protocol. This will be the focus of future work. As the use of test votes is likely to have a negative impact on usability, we will make further improvements in this regard. One proposal is for the voter to enter some unique information known only to him. Verifying the presence of this information at a later stage assures him of the integrity of his submitted vote. Proposals will be made with the objective of balancing security and the expectations and abilities of the human peers in the ceremony.

VI. ACKNOWLEDGEMENT

Support by CAPES foundation Brazil.

REFERENCES

[1] D. Chaum and et al., "Scantegrity ii: End-to-end verifiability for optical scan election systems using invisible ink confirmation codes," ser. EVT'08. USENIX Association, 2008.

[2] C. Burton, , and et al., "Using Prêt à Voter in Victorian State Elections," ser. EVT/WOTE'12. USENIX Association, 2012.

[3] B. Adida, "Helios: Web-based Open-Audit Voting," in Proceedings of the 17th Symposium on Security. Usenix Association, 2008, pp. 335 – 348.

[4] B. Adida, O. De Marneffe, O. Pereira, and J.-J. Quisquater, "Electing A University President using Open-Audit Voting: Analysis of Real-World Use of Helios," ser. EVT/WOTE'09. Usenix Association, 2009.

[5] Princeton, "Princeton Undergraduate Elections," accessed on 30th June, 2015. [Online]. Available: https://princeton.heliosvoting.org/

[6] IACR, "IACR Board of Directors 2013 Election and Referendum on Bylaws Amendments," accessed on 30th June, 2015. [Online]. Available: https://vote.heliosvoting.org/helios/elections/b36cbf0c-250a-11e3-89f4-46d2afa631be/view

[7] F. Karayumak, M. Kauer, M. M. Olembo, and M. Volkamer, "Usability Analysis of Helios - An Open Source Verifiable Remote Electronic Voting System," ser. EVT/WOTE'12. USENIX Association, 2011.

[8] F. Karayumak, M. Kauer, M. M. Olembo, T. Volk, and M. Volkamer, "User Study of the Improved Helios Voting System Interfaces," in STAST, 2011.

[9] M. A. Sasse and I. Flechais, "Usable Security: Why Do We Need It? How Do We Get It?" in Security and Usability: Designing Secure Systems That People Can Use. O'Reilly, 2005.

[10] C. Ellison, "Ceremony Design and Analysis," Cryptology ePrint Archive, Report 2007/399, 2007.

[11] M. C. Carlos, J. Martina, G. Price, and R. F. Custodio, "An Updated Threat Model for Security Ceremonies," in ACM Symposium on Applied Computing. ACM, 2013.

[12] D. Dolev and A. C. Yao, "On the Security of Public Key Protocols," IEEE Transactions on Information Theory, vol. 29, 1983.

[13] D. Demirel, J. Van De Graaf, and R. Araújo, "Improving Helios with Everlasting Privacy Towards the Public," ser. EVT/WOTE'12. USENIX Association, 2012.

[14] V. Cortier, D. Galindo, S. Glondu, and M. Izabachene, "A Generic Construction for Voting Correctness at Minimum Cost - Application to Helios," Cryptology ePrint Archive, Report 2013/177, 2013.

[15] V. Cortier and B. Smyth, "Attacking and Fixing Helios: An Analysis of Ballot Secrecy," ser. CSF '11. IEEE Computer Society, 2011.

[16] G. Tsoukalas, K. Papadimitriou, P. Louridas, and P. Tsanakas, "From Helios to Zeus," EVT/WOTE'13.

[17] B. Adida, "Helios: Web-based open-audit voting."

[18] R. M. Needham and M. D. Schroeder, "Using Encryption for Authentication in Large Networks of Computers," ACM Press, 1978.

[19] S. Neumann, J. Budurushi, and M. Volkamer, Analysis of Security and Cryptographic Approaches to Provide Secret and Verifiable Electronic Voting. IGI Global, 2013.

[20] L. Langer, A. Schmidt, J. Buchmann, and M. Volkamer, "A Taxonomy Refining the Security Requirements for Electronic Voting: Analyzing Helios as a Proof of Concept," in ARES'10. IEEE, 2010.

[21] R. Küsters, T. Truderung, and A. Vogt, "Clash Attacks on the Verifiability of E-Voting Systems," in IEEE Symposium, 2012, pp. 395–409.

[22] D. A. Norman, The Design of Everyday Things. Basic Books, Inc., 2002.

 ISBN: 978-1-61208-427-5