Proving Coercion-Resistance of Scantegrity II[⋆]

Ralf Küsters, Tomasz Truderung, and Andreas Vogt

University of Trier
{kuesters,truderun,vogt}@uni-trier.de

Abstract. By now, many voting protocols have been proposed that, among others, are designed to achieve coercion-resistance, i.e., resistance to vote buying and voter coercion. Scantegrity II is among the most prominent and successful such protocols in that it has been used in several elections. However, almost none of the modern voting protocols used in practice, including Scantegrity II, has undergone a rigorous cryptographic analysis.

In this paper, we prove that Scantegrity II enjoys an optimal level of coercion-resistance, i.e., the same level of coercion-resistance as an ideal voting protocol (which merely reveals the outcome of the election), except for so-called forced abstention attacks. This result is obtained under the (necessary) assumption that the workstation used in the protocol is honest.

Our analysis is based on a rigorous cryptographic definition of coercion-resistance we recently proposed. We argue that this definition is in fact the only existing cryptographic definition of coercion-resistance suitable for analyzing Scantegrity II. Our case study should encourage and facilitate rigorous cryptographic analysis of coercion-resistance also for other voting protocols used in practice.

1 Introduction

By now, many voting protocols have been proposed that are designed to achieve (various forms of) verifiability [6, 8] and receipt-freeness/coercion-resistance [1]. Among the first paper-based protocols that try to achieve these properties are protocols by Chaum [3], Neff [16], and Prêt à Voter [18]. Scantegrity II is among the most prominent and successful such protocols in that it has been used in several elections [4]. Intuitively, verifiability means that voters can check whether the result of the election is correct. For this purpose, voters are typically given some kind of receipt and besides the result of the election additional data is published. However, this might open the door to vote buying and voter coercion. Therefore, coercion-resistance, i.e., resistance to vote buying and voter coercion, is required as well. While the voting schemes are quite complex and coercion-resistance is a very intricate property, almost none of the modern voting protocols used in practice, including Scantegrity II, has undergone a rigorous cryptographic analysis (see Section 5). The main goal of this work is therefore to provide such an

[⋆] This work was partially supported by the DFG under Grant KU 1434/6-1.

analysis for a practical and non-trivial voting system, namely Scantegrity II. We believe that our case study will encourage and facilitate rigorous cryptographic analysis of coercion-resistance also for other voting protocols used in practice.

Contribution of this Paper. In this paper, we show that Scantegrity II provides an optimal level of coercion-resistance, i.e., the same level of coercion-resistance as an ideal voting protocol (which merely reveals the outcome of the election), except for so-called forced abstention attacks: We assume the coercer to be quite powerful in that he can see the receipts of all voters, and hence, a coercer can force voters to abstain from voting. Our analysis assumes that the workstation used by Scantegrity II is honest. This assumption, as we will show, is necessary for the system to be coercion-resistant.

Our analysis is based on a rigorous cryptographic definition of coercion-resistance we recently proposed [13]. Compared to other cryptographic definitions, e.g., [10, 15, 19], our definition is quite simple and intuitive and promises to be widely applicable. We argue in Section 4.1 that other cryptographic definitions are unsuitable for the analysis of Scantegrity II.

Structure of this Paper. In the following section, we recall the definition of coercion-resistance from [13]. In Section 3, we describe the Scantegrity II voting system and present a formal specification. The analysis of Scantegrity II is then presented in Section 4. Related work is discussed in Section 5. Further details of our analysis are provided in the appendix.

2 Coercion-Resistance

In this section, we briefly recall the definition of coercion-resistance from [13] as well as the level of coercion-resistance an ideal voting protocol has, as this is used in Section 3. First, we introduce some notation and terminology.

2.1 Preliminaries

As usual, a function f from the natural numbers to the real numbers is *negligible* if for every $c > 0$ there exists ℓ_0 such that $f(\ell) \leq \frac{1}{\ell^c}$ for all $\ell > \ell_0$. The function f is *overwhelming* if the function $1 - f(\ell)$ is negligible. Let $\delta \in [0, 1]$. The function f is *δ-bounded* if f is bounded by δ plus a negligible function, i.e., for every $c > 0$ there exists ℓ_0 such that $f(\ell) \leq \delta + \frac{1}{\ell^c}$ for all $\ell > \ell_0$.

Our modeling will be based on a computational model similar to models for simulation-based security (see [11] and Appendix A), in which *interactive Turing machines (ITMs)* communicate via tapes. The details of this model are not necessary to be able to follow the rest of the paper. However, we fix some notation and terminology. A *system* $\mathcal{S}$ of ITMs is a multi-set of ITMs, which we write as $\mathcal{S} = M_1 \parallel \cdots \parallel M_l$, where $M_1, \ldots, M_l$ are ITMs. If $\mathcal{S}_1$ and $\mathcal{S}_2$ are systems of ITMs, then $\mathcal{S}_1 \parallel \mathcal{S}_2$ is a system of ITMs, provided that $\mathcal{S}_1$ and $\mathcal{S}_2$ are connectible w.r.t. their interfaces (external tapes). Clearly, a run of a system $\mathcal{S}$ is uniquely determined by the random coins used by the ITMs in $\mathcal{S}$. We assume that a system of ITMs has at most one ITM with a special output tape $\mathsf{decision}$. For a system $\mathcal{S}$ of ITMs and a security parameter ℓ, we write $\mathsf{Pr}[\mathcal{S}^{(\ell)} \mapsto 1]$ to denote the probability that $\mathcal{S}$ outputs 1 (on tape $\mathsf{decision}$) in a run with security parameter ℓ.

A *property* of a system $\mathcal{S}$ is a subset of runs of $\mathcal{S}$. For a property γ of $\mathcal{S}$, we write $\mathsf{Pr}[\mathcal{S}^{(\ell)} \mapsto \gamma]$ to denote the probability that a run of $\mathcal{S}$, with security parameter ℓ, belongs to γ.

2.2 Voting Protocols

A *voting protocol* P specifies the programs (actions) carried out by honest voters and honest voting authorities, such as honest registration tellers, tallying tellers, bulletin boards, etc.

A voting protocol P, together with certain parameters, induces an *election system* $S = P(k, m, n, \boldsymbol{p})$. The parameters are as follows: k denotes the number of choices an honest voter has in the election, e.g., the number of candidates a voter can vote for, apart from abstaining from voting. By m we denote the total number of voters and by n, with $n \leq m$, the number of honest voters. Honest voters follow the programs as specified in the protocol. The actions of dishonest voters and dishonest authorities are determined by the coercer, and hence, these participants can deviate from the protocol specification in arbitrary ways. The parameter n is made explicit since it is crucial for the level of coercion-resistance a system guarantees. One can also think of n as the minimum number of voters the coercer may not corrupt. The vector $\boldsymbol{p} = p_0, \ldots, p_k$ is a probability distribution on the possible choices, i.e., $p_0, \ldots, p_k \in [0, 1]$ and $\sum_{i=0}^{k} p_i = 1$. Honest voters will abstain from voting with probability p_0 and vote for candidate i with probability p_i, $1 \leq i \leq k$. This distribution is made explicit, because it is realistic to assume that the coercer knows this distribution (e.g., from opinion polls), and hence, uses it in his strategy, and because the specific distribution is crucial for the level of coercion-resistance of a system.

An election system $S = P(k, m, n, \boldsymbol{p})$ specifies (sets of) ITMs for all participants, i.e., honest voters and authorities, the coercer (who subsumes all dishonest voters and dishonest authorities), and the coerced voter: (i) There are ITMs, say $S_1, \ldots, S_l$, for all honest voting authorities. These ITMs run the programs as specified by the voting protocol. (ii) There is an ITM S_{v_i}, $i \in \{1, \ldots, n\}$, for each of the honest voters. Every such ITM first makes a choice according to the probability distribution $\boldsymbol{p}$. Then, if the choice is not to abstain, it runs the program for honest voters according to the protocol specification with the candidate chosen before. (iii) The coercer is described by a set C_S of ITMs. This set contains all (probabilistic polynomial-time) ITMs, and hence, all possible coercion strategies the coercer can carry out. These ITMs are only constrained in their interface to the rest of the system. Typically, the ITMs can directly use the interface of dishonest voters and authorities. They can also communicate with the coerced voter and have access to all public information (e.g., bulletin boards) and possibly (certain parts of) the network. The precise interface of the ITMs in C_S depends on the specific protocol and the assumptions on the power of the coercer. (iv) Similarly, the coerced voter is described by a set V_S of ITMs. Again, this set contains all (probabilistic polynomial-time) ITMs. This set represents all the possible programs the coercer can ask the coerced voter to run as well as all counter-strategies the coerced voter can run (see Section 2.3 for more explanation). The interface of these ITMs is typically the interface of an honest voter plus an interface for communication with the coercer. In particular, the set V_S contains what we call a *dummy strategy* dum which simply forwards all the messages between the coercer and the interface the coerced voter has as an honest voter.

Given an election system $S = P(k, m, n, \boldsymbol{p})$, we denote by e_S the system of ITMs containing all honest participants, i.e., $\mathsf{e}_S = (S_{\mathsf{v}_1} \parallel \ldots \parallel S_{\mathsf{v}_n} \parallel S_1 \parallel \ldots \parallel S_l)$. A system $(c \parallel v \parallel \mathsf{e}_S)$ of ITMs, with $c \in C_S$ and $v \in V_S$, is called an *instance of S*. We often implicitly assume a scheduler, modeled as an ITM, to be part of the system(see Appendix A for a concrete example). Its role is to make sure that all components of the system are scheduled in a fair way, e.g., all voters get a chance to vote. For simplicity of notation, we do not state the scheduler explicitly. We define a *run of S* to be a run of some instance of S.

For an election system $S = P(k, m, n, \boldsymbol{p})$, we denote by $\Omega_1 = \{0, \ldots, k\}^n$ the set of all possible combinations of choices made by the honest voters, with the corresponding probability distribution μ_1 derived from $\boldsymbol{p} = p_0, p_1, \ldots, p_k$. All other random bits used by ITMs in an instance of S, i.e., all other random bits used by honest voters as well as all random bits used by honest authorities, the coercer, and the coerced voter, are uniformly distributed. We take μ_2 to be this distribution over the space Ω_2 of random bits. Formally, this distribution depends on the security parameter. We can, however, safely ignore it in the notation without causing confusion. We define $\Omega = \Omega_1 \times \Omega_2$ and $\mu = \mu_1 \times \mu_2$, i.e., μ is the product distribution obtained from μ_1 and μ_2. For an event φ, we will write $\mathsf{Pr}_{\omega_1,\omega_2 \leftarrow \Omega}[\varphi]$, $\mathsf{Pr}_{\omega_1,\omega_2}[\varphi]$, or simply $\mathsf{Pr}[\varphi]$ to denote the probability $\mu(\{(\omega_1, \omega_2) \in \Omega : \varphi(\omega_1, \omega_2)\})$. Similarly, $\mathsf{Pr}_{\omega_1 \leftarrow \Omega_1}[\varphi]$ or simply $\mathsf{Pr}_{\omega_1}[\varphi]$ will stand for $\mu_1(\{\omega_1 \in \Omega_1 : \varphi(\omega_1)\})$; analogously for $\mathsf{Pr}_{\omega_2 \leftarrow \Omega_2}[\varphi]$.

A *property* of an election system $S = P(k, m, n, \boldsymbol{p})$ is defined to be a class γ of properties containing one property γ_T for each instance T of S. We will write $\mathsf{Pr}\,[T \mapsto \gamma]$ to denote the probability $\mathsf{Pr}\,[T \mapsto \gamma_T]$.

2.3 Defining Coercion-Resistance

We can now recall the definition of coercion-resistance from [13] (see [13] for more explanation). In what follows, let P be a voting protocol and $S = P(k, m, n, \boldsymbol{p})$ be an election system for P.

The definition of coercion-resistance assumes that a coerced voter has a certain goal γ that she would try to achieve in absence of coercion. Formally, γ is a property of S. If, for example, γ is supposed to express that the coerced voter wants to vote for a certain candidate, then γ would contain all runs in which the coerced voter voted for this candidate and this vote is in fact counted.

In the definition of coercion-resistance the coercer demands full control over the voting interface of the coerced voter, i.e., the coercer wants the coerced voter to run the dummy strategy dum (which simply forwards all the messages between the coercer and the interface the coerced voter has as an honest voter) instead of the program an honest voter would run.

Now, for a protocol to be coercion-resistant the definition requires that there exists a *counter-strategy* $\tilde{v}$ that the coerced voter can run instead of dum such that (i) the coerced voter achieves her own goal γ, with overwhelming probability, by running $\tilde{v}$ and (ii) the coercer is not able to distinguish whether the coerced voter runs dum or $\tilde{v}$. If such a counter-strategy exists, then it indeed does not make sense for the coercer to try to influence a voter in any way, e.g., by offering money or threatening the voter, at least not from a technical point of view:[1] Even if the coerced voter tries to sell her vote, the coercer is not able to tell whether she is actually following the coercer's instructions or just trying to achieve her own goal by running the counter-strategy. For the same reason, the coerced voter is safe even if she wants to achieve her goal and therefore runs the counter-strategy.

The formal definition of coercion-resistance is the following:

Definition 1. Let P be a protocol and $S = P(k, m, n, \boldsymbol{p})$ be an election system. Let $\delta \in [0, 1]$, and γ be a property of S. The system S is *δ-coercion-resistant w.r.t.* γ, if there exists $\tilde{v} \in V_S$ such that for all $c \in C_S$ we have:

(i) $\mathsf{Pr}[(c \parallel \tilde{v} \parallel \mathsf{e}_S)^{(\ell)} \mapsto \gamma]$ is overwhelming, as a function of the security parameter.

(ii) $\mathsf{Pr}[(c \parallel \mathsf{dum} \parallel \mathsf{e}_S)^{(\ell)} \mapsto 1] - \mathsf{Pr}[(c \parallel \tilde{v} \parallel \mathsf{e}_S)^{(\ell)} \mapsto 1]$ is δ-bounded, as a function of the security parameter.

Condition (i) says that by running the counter-strategy $\tilde{v}$ the coerced voter achieves her goal with overwhelming probability, no matter which coercion-strategy the coercer performs. Condition (ii) captures that the coercer is unable to distinguish whether the coerced voter runs dum or $\tilde{v}$, i.e., whether the coerced voter follows the instructions of the coercer or simply runs the counter-strategy, and hence, tries to achieve her own goal. As we will see below, replacing "δ-bounded" by "negligible" would be too strong a condition.

As further discussed in [13], it suffices to interpret dum ($\tilde{v}$) in Definition 1 to be a single coerced voter since this covers coercion-resistance for the case of multiple coerced voters.

2.4 The Level of Coercion-Resistance of the Ideal Protocol

The *ideal protocol* simply collects all votes of the voters and outputs the correct result. In this section, we recall the level of coercion-resistance of this protocol, as established in [13]. This will be used to determine the level of coercion-resistance of Scantegrity II in Section 3.

We consider the goal γ_i of the coerced voter, for $i \in \{1, \ldots, k\}$, defined as follows: A run belongs to γ_i if, whenever the coerced voter has indicated

[1] Of course, voters can be influenced psychologically.

her candidate to the voting authority, she has successfully voted for the i-th candidate. Note that this implies that if the coerced voter is not instructed by the coercer to vote, and hence, effectively wants the coerced voter to abstain from voting, the coerced voter does not have to vote in order to fulfill γ_i. In other words, by γ_i abstention attacks are not prevented. As discussed in [13], for the ideal protocol a stronger goal, which excludes abstention attacks, can be achieved by the coerced voter. However, such a goal would be too strong for Scantegrity II, as abstention attacks are not prevented (see Section 4). In order to be able to reduce the analysis of Scantegrity II to the ideal case, we therefore consider γ_i here, instead of the stronger goal.

Since the coercer knows the votes of dishonest voters (the coercer subsumes these voters), he can simply subtract these votes from the final result and obtain what we will call the *pure result* of the election. The pure result only depends on the votes of the n honest voters and the coerced voter. Hence, a pure result is a tuple $\boldsymbol{r} = (r_0, \ldots, r_k)$ of non-negative integers such that $r_0 + \cdots + r_k = n + 1$, where r_i, for $i \in \{1, \ldots, k\}$, is the number of votes for the i-th candidate and r_0 denotes the number of voters who abstained from voting. We will denote the set of pure results by Res.

To state the level $\delta = \delta_{min}(k, n, \boldsymbol{p})$ of coercion-resistance of the ideal protocol, as established in [13], we use the probability $A^i_{\boldsymbol{r}}$ that the choices made by the honest voters and the coerced voter yield the pure result $\boldsymbol{r} = (r_0, \ldots, r_k)$, given that the coerced voter votes for the i-th candidate. Let $r'_j = r_j$ for $j \neq i$ and $r'_j = r_i - 1$. It is easy to see that

$$A^i_{\boldsymbol{r}} \;=\; \frac{n!}{r'_0! \cdots r'_k!} \cdot p_0^{r'_0} \ldots p_k^{r'_k} \;=\; \frac{n!}{r_0! \cdots r_k!} \cdot p_0^{r_0} \ldots p_k^{r_k} \cdot \frac{r_i}{p_i}.$$

The intuition behind the definition of $\delta_{min}(k, n, \boldsymbol{p})$ is the following: If the coercer wants the coerced voter to vote for j and the coerced voter wants to vote for i, for some $i, j \in \{1, \ldots, k\}$, then the best strategy of the coercer to distinguish whether the coerced voter has voted for j or i is to accept a run if the pure result $\boldsymbol{r}$ of the election in this run is such that $A^i_{\boldsymbol{r}} \leq A^j_{\boldsymbol{r}}$. Let $M^*_{i,j} = \{\boldsymbol{r} \in \mathit{Res} : A^i_{\boldsymbol{r}} \leq A^j_{\boldsymbol{r}}\}$ be the set of those results, for which—according to his best strategy—the coercer should accept the run. Now, we are ready to define the constant $\delta^i_{min}(n, k, \boldsymbol{p})$, which is shown to be optimal in [13]:

$$\delta^i_{min}(n, k, \boldsymbol{p}) = \max_{j \in \{1,\ldots,k\}} \sum_{\boldsymbol{r} \in M^*_{i,j}} (A^j_{\boldsymbol{r}} - A^i_{\boldsymbol{r}}).$$

Figure 1, which we took from [13], shows $\delta^i_{min}(n, k, \boldsymbol{p})$ for some selected cases. As illustrated, the level of coercion-resistance decreases with the number of candidates and increases with the number of honest voters. It also depends on the probability distribution on candidates. For example, in case of 5 candidates and 10 honest voters, δ is about 0.6. So, since δ is optimal, there exists a coercion strategy c such that the probability of the coercer accepting the run (i.e. returning 1) is 60% higher in case the coerced voter performs dum, compared to performing $\tilde{v}$. This gives strong incentives for the coerced voter to follow the instructions of the coercer, i.e., run dum: In case the coerced voter is threatened by the coercer, chances of being punished would be reduced significantly. In case the coerced voter wants to sell her vote, chances of being payed increase significantly.

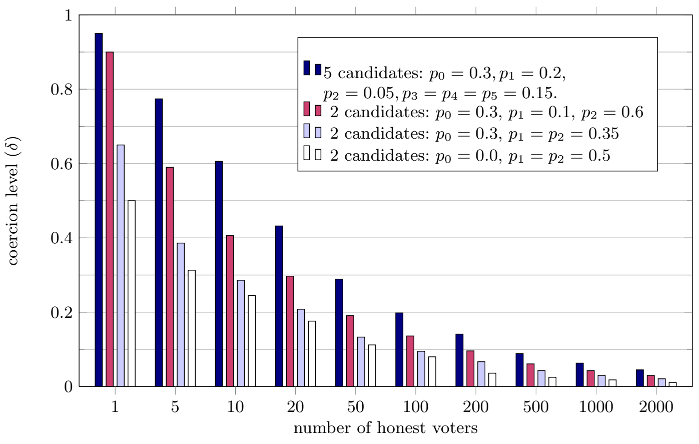

Fig. 1. Level of coercion-resistance (δ) for the ideal protocol. The goal of the coerced voter is, in each case, to vote for candidate 1.

3 Scantegrity II

In this section, we first give an informal description of the Scantegrity II system [4]. We then provide a formal specification as an election system, as introduced in Section 2.2. We will denote the Scantegrity II system by $\mathsf{P}_{\mathsf{Sct}}$.

3.1 Informal Description

In addition to the voters, the participants in this system are the following: (i) A *workstation (WSt)*, which is the main component in the voting process. The workstation controls a *bulletin board* which the workstation uses for broadcasting messages; everybody has read access to this bulletin board. A scanner and a *pseudo random number generator (PRNG)* are also part of the workstation. (ii) Some number of *auditors* $aud_1, \ldots, aud_t$ who will contribute randomness in a distributed way used for randomized partial checking (RPC). (iii) A number of clerks $cl_1, \ldots, cl_r$ who have shares of a secret seed that is given to the PRNG; the length of the seed is determined by the security parameter.

The election consists of three phases described below: initialization, voting, and tallying.

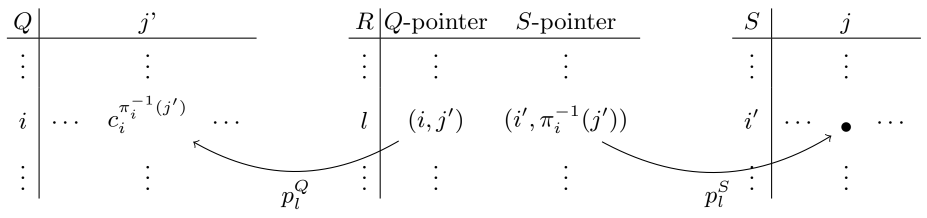

Fig. 2. Q-, R-, and S-table

Initialization phase. In this phase, the election officials $cl_1, \ldots, cl_r$ secret-share a seed and input this seed to the PRNG. The pseudo-random string produced by the PRNG is the only source of randomness of the workstation. Using this string, the workstation creates a so-called P-table, which consists of $k \cdot s$ (pseudo-random) confirmation codes $\{c_i^j\}_{\substack{i=1,\ldots,s \\ j=1,\ldots,k}}$ of constant length, where s is at least twice as high as the number of voters and k is the number of candidates. This table will never be published. For every row $i \in \{1, \ldots, s\}$ in the P-table, a ballot is printed with the serial number i and the confirmation codes c_i^j written in invisible ink next to the respective candidate name $j \in \{1, \ldots, k\}$. The workstation also creates a Q-table $\{c_i^{\pi_i^{-1}(j)}\}_{\substack{i=1,\ldots,s \\ j=1,\ldots,k}}$ obtained from the P-table by permuting cells with a pseudo-random permutation π_i in each row i. Next, the so-called S-table of size $k \cdot s$ is created. This table is initially empty and will be used to mark positions corresponding to the candidates chosen by the voters. Furthermore, another table, the R-table is created. The R-table consists of two columns, one column for Q-pointers p_l^Q and one column for S-pointers p_l^S, for $l = 1, \ldots, (s \cdot k)$. These pointers are just indices of the respective table and are (supposed to be) pseudo-randomly generated in a way that for every cell $(i, j) \in \{1, \ldots, s\} \times \{1, \ldots, k\}$ of the Q-table (S-table), there is exactly one Q-pointer $p_l^Q = (i, j)$ (one S-pointer $p_l^S = (i, j)$) pointing to that cell. Moreover, for every l, if $p_l^Q = (i, j')$ and $p_l^S = (i', j)$, then $j = \pi_i^{-1}(j')$, i.e., the S-pointer next to a Q-pointer pointing to a cell with confirmation code $c_i^{\pi_i^{-1}(j)}$ for candidate $j = \pi_i^{-1}(j')$, points to a cell in the j-th column of the S-table (see Figure 2). The workstation commits on every entry in the Q- and R-table and publishes these commitments. The workstation uses a perfectly-hiding and computationally-binding commitment scheme (e.g., Pedersen commitments).

Voting phase. In this phase the voter asks for either one or two ballots and a decoder pen which she can use to reveal the codes written in invisible ink. If she takes two ballots, she chooses one ballot to audit, which means that all codes are revealed and the workstation has to open all the corresponding commitments in the Q- and R-table. Intuitively, because of this check, a workstation that cheats by producing wrong ballots or wrong tables is detected with high probability.

The other ballot is used for voting: The voter unveils exactly the code next to the candidate she wants to vote for and may note down that code. This code constitutes the voter's receipt. Unveiling another code would invalidate the ballot. Unveiling the code darkens the space next to the candidate, which can be detected by an optical scanner. The voter has her ballot scanned by a scanner, which records the candidate chosen by the voter together with the ballot serial number.

Tallying phase. In this last phase, the election officials publish a list of all voters that voted and the tally given by the optical scanners. Furthermore, the workstation uses the P-table to reconstruct, for every recorded pair (i, j) of ballot serial number i and candidate j, the confirmation code c_i^j. The commitment to that code in the Q-table is then opened, i.e., the commitment on the value of the cell (i, j') of the Q-table, with $\pi_i^{-1}(j') = j$. Furthermore, the corresponding cells in the R- and S-table are flagged: the election officials flag (publish) the index l of the R-table such that $p_l^Q = (i, j')$ and flag (publish) the index p_l^S of the S-table. Finally, for each row l of the R-table, either the commitment on the Q-pointer p_l^Q or on the S-pointer p_l^S is opened, depending on a publicly verifiable coin flip, provided by the auditors. Intuitively, this auditing should prevent the workstation from flagging entries in the S-table in a way that does not correspond to the actual votes.

Now, the result can be easily computed from the publicly available information: the number of votes for candidate j is the number of flagged cells in the j-th column of the S-table.

3.2 Modeling and Security Assumptions

The formal specification of Scantegrity II as an election system in the sense of Section 2.2 is straightforward. However, we highlight some modeling issues and, most importantly, state our security assumptions. A detailed model is provided in Appendix A.

Voting Authorities. We assume that the workstation, including the PRNG and the scanner, as well as at least one clerk cl_i $(i = 1, \ldots, r)$ are honest; the auditors may all be dishonest. These assumptions are necessary for Scantegrity II to be coercion-resistance: The dishonest workstation could reveal all votes to the coercer. A dishonest PRNG could leak the pseudo-random string to the coercer, allowing the coercer to deduce the candidate-code-pairs that appear on receipts, and hence, read off from a receipt how a voter voted. If all clerks were dishonest, they could leak the seed for the PRNG, leading to the same problem as in the case of a dishonest PRNG.

Honest voters. Honest voters act as described in Section 2.2: first, make a choice according to the probability distribution $\boldsymbol{p}$ and then, if the choice is not to abstain from voting, follow the procedure described for the voting phase. We assume an untappable channel from the voter to the workstation. This models that voters vote in a voting booth. After the voting phase is finished for all voters, voters provide the coercer with their receipt (if any); they might for example give their receipts to an organization to ask it to verify the correctness of the voting process w.r.t. her receipt or to publish it on some bulletin board. Hence, the coercer is provided with the receipts of all voters, which makes the coercer quite strong. In particular, the coercer knows whether or not a voter voted, making it impossible to prevent abstention attacks. The assumption that the receipts are revealed after the voting phase is reasonable. Also, the (presumably small) fraction of honest voters for which the coercer manages to get hold of the receipt earlier, could be considered to be dishonest.

The coerced voter. A coerced voter has the same interface as an honest voter (including the untappable channel to the workstation), plus a channel to (freely) communicate with the coercer. Note that the coercer does not have direct access to the untappable channel, only via the coerced voter. In particular, while the coerced voter could be on the phone with the coercer all the time, the coerced voter can lie about what she sees and does in the voting booth. (This excludes taking pictures or videos in the booth, unless the coerced voter can modify these pictures and videos on-the-fly.)

The coercer. The coercer subsumes all dishonest parties, i.e., dishonest voters and authorities. In fact, these parties are considered to be part of the coercer. In a run of the system the coercer can see the following: (v1) his random coins, (v2) all published messages (on the bulletin board), both in the initialization phase and the tallying phase, (v3) receipts of all honest voters, as explained above, and (v4) all messages received from the coerced voter, including her receipt.

4 Analysis of Scantegrity II

In this section, we show that the Scantegrity II system, as specified in Section 3, enjoys the same level of coercion-resistance as the ideal protocol, unlike other protocols, for instance, ThreeBallot, analyzed in [13].

4.1 The Main Result

We prove the following theorem, where we will consider goals γ_i of the coerced voter, for $i \in \{1, \ldots, k\}$, as described in Section 2.4.

Theorem 1. *Let $S = \mathsf{P}_{\mathsf{Sct}}(k, m, n, \boldsymbol{p})$. Then S is δ-coercion-resistant with respect to γ_i, where $\delta = \delta^i_{min}(n, k, \boldsymbol{p})$. Moreover, δ is optimal, i.e., for every $\delta' < \delta$ the system $\mathsf{P}_{\mathsf{Sct}}(k, m, n, \boldsymbol{p})$ is not δ'-coercion-resistant w.r.t. γ_i.*

The optimality of δ directly follows from the fact that, as mentioned, $\delta^i_{min}(n, k, \boldsymbol{p})$ is the optimal level of coercion-resistance of the ideal voting protocol.

We note that none of the other existing cryptographic definitions of coercion-resistance is suitable for the analysis of Scantegrity II: The definition by Juels et al. [10] is tailored towards voting in a public-key setting, with protocols having a specific structure. Scantegrity II does not fall into their class of voting protocols. The definition by Moran and Naor [15] is simulation-based, and hence, suffers from the so-called commitment problem. Due to this problem, the definition by Moran and Naor would reject Scantegrity II as insecure. The definition by Teague et al. [19] is intended to be used for ideal voting functionalities, which again excludes Scantegrity II.

4.2 Proof of the Main Result

The remainder of this section is devoted to the proof of Theorem 1. First, we define the counter-strategy $\tilde{v}$ of the coerced voter: $\tilde{v}$ coincides with the dummy strategy dum, with the exception that $\tilde{v}$ votes for candidate i, i.e., the coerced voter reveals the code next to candidate i, if the coercer instructs the coerced voter to vote for some candidate j.

Clearly, if the coerced voter runs the counter-strategy $\tilde{v}$, then condition (i) of Definition 1 is satisfied for every $c \in C_S$. Note that if the coercer does not instruct the coerced voter to vote for some candidate j (abstention attack), then following the counter-strategy the coerced voter abstains from voting, which is in accordance with γ_i.

It remains to prove condition (ii) of Definition 1. For this purpose, let us fix a program c of the coercer. We need to prove that $\mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1] \leq \delta$, where $T = (\mathsf{dum} \parallel c \parallel \mathsf{e}_S)$ and $\tilde{T} = (\tilde{v} \parallel c \parallel \mathsf{e}_S)$. A simple reduction allows us to replace the pseudo-random bit string, produced by the PRNG, by a real random bit string (see Appendix B.1). In what follows, we therefore assume that the workstation is given a real random bit string. The rest of the proof consists of two parts, a cryptographic and a combinatorial part, following a similar structure as a proof carried out in [13] for the the Bingo Voting system (see also Section 5). The cryptographic part is Lemma 1. Using Lemma 1, the combinatorial part consists in a reduction to the ideal case (see Section 2.4). We can then use the results for the ideal protocol.

As introduced in Section 2.2, by $\omega_1 \in \Omega_1$ we denote a vector of choices made by the honest voters and by $\omega_2 \in \Omega_2$ we denote all the remaining random coins of a system. A *pure result* $\boldsymbol{r} = (r_0, \ldots, r_k)$ is defined as in Section 2.4. We denote by ρ a view of the coercer, as described in Section 3.2, (v1)–(v4). We will denote the pure result determined by a view ρ of the coercer by $\mathrm{res}(\rho)$. A pure result determined by ω_1 and the choice j of the coerced voter will be denoted by $\mathrm{res}(\omega_1, j)$.

For a coercer view ρ in a run of the system, we denote by $f(\rho)$ the candidate the coercer wants the coerced voter to vote for; if the coercer does not instruct the coerced voter to vote, then $f(\rho)$ is undefined. Note that the coercer has to provide the coerced voter with $f(\rho)$ before the end of the voting phase. All messages the coercer has seen up to this point only depend on ω_2 and are independent of the choices made by honest voters. Therefore, we sometimes write $f(\omega_2)$ for the candidate the coercer wants the coerced voter to vote for in runs that use the random coins ω_2.

The coercer can derive from his view which voters abstained from voting as he sees the receipts of the voters that successfully voted. Given a view ρ of the coercer, we denote by $\mathrm{abst}(\rho)$ the set of voters who did not vote successfully, among the honest voters and the coerced voter; the number of such voters is referred to by $r_0(\rho) = |\mathrm{abst}(\rho)|$. Below we will consider only views ρ such that $f(\rho)$ is defined. In this case the set $\mathrm{abst}(\rho)$ and the number $r_0(\rho)$ depend only on ω_1. We will therefore also write $\mathrm{abst}(\omega_1)/r_0(\omega_1)$.

For a coercer view ρ in T, where the coerced voter runs the dummy strategy, let φ_ρ be a predicate over Ω_1 such that $\varphi_\rho(\omega_1)$ holds iff $\mathrm{res}(\omega_1, f(\rho)) = \mathrm{res}(\rho)$ and $\mathrm{abst}(\omega_1) = \mathrm{abst}(\rho)$, i.e., the choices ω_1 of the honest voters are consistent with the view of the coercer, as far as the result of the election and the set of abstaining voters is concerned. Analogously, for a coercer view ρ in $\tilde{T}$, where the coerced voter runs the counter-strategy, we define that $\tilde{\varphi}_\rho(\omega_1)$ holds iff $\mathrm{res}(\omega_1, i) = \mathrm{res}(\rho)$ and $\mathrm{abst}(\omega_1) = \mathrm{abst}(\rho)$.

For a coercer view ρ, by $T(\omega_1, \omega_2) \mapsto \rho$, or simply $T \mapsto \rho$, we denote the fact that the system T, when run with ω_1, ω_2, produces the view ρ (similarly for $\tilde{T}$). For a set M of views, we write $T(\omega_1, \omega_2) \mapsto M$ if $T(\omega_1, \omega_2) \mapsto \rho$ for some $\rho \in M$.

The following lemma is the key fact used in the proof of Theorem 1. It constitutes the cryptographic part of the proof of Theorem 1.

Lemma 1. *Let ρ be a coercer view such that $f(\rho)$ is defined. Let ω_1^ρ and $\tilde{\omega}_1^\rho$ be some fixed elements of Ω_1 such that $\varphi_\rho(\omega_1^\rho)$ and $\tilde{\varphi}_\rho(\tilde{\omega}_1^\rho)$, respectively. Then, the following equations hold true:*

$$\mathsf{Pr}[T \mapsto \rho] = \mathsf{Pr}_{\omega_1}[\varphi_\rho(\omega_1)] \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^\rho, \omega_2) \mapsto \rho] \tag{1}$$

$$\mathsf{Pr}[\tilde{T} \mapsto \rho] = \mathsf{Pr}_{\omega_1}[\tilde{\varphi}_\rho(\omega_1)] \cdot \mathsf{Pr}_{\omega_2}[\tilde{T}(\tilde{\omega}_1^\rho, \omega_2) \mapsto \rho] \tag{2}$$

$$\mathsf{Pr}_{\omega_2}[T(\omega_1^\rho, \omega_2) \mapsto \rho] = \mathsf{Pr}_{\omega_2}[\tilde{T}(\tilde{\omega}_1^\rho, \omega_2) \mapsto \rho] \ . \tag{3}$$

Intuitively, the lemma says that the view of the coercer is information-theoretically independent of the choices of honest voters and the coerced voter as long as these choices are consistent with the result of the election given in this view.

The proof of Lemma 1 (see Appendix B.2) heavily depends on the details of Scantegrity II. In contrast, using this lemma, the reduction to the ideal case for the combinatorial part of the proof of Theorem 1 is now generic and quite independent of Scantegrity II, making this proof technique a useful tool also for the analysis of other protocols.

For the reduction, we first observe that if $f(\rho)$ is defined, then we have:

$$\begin{aligned}\mathsf{Pr}_{\omega_1}[\varphi_\rho(\omega_1)] &= \mathsf{Pr}_{\omega_1}[\mathrm{res}(\omega_1, f(\rho)) = \mathrm{res}(\rho)] \cdot \\ &\quad \cdot \mathsf{Pr}_{\omega_1}[\mathrm{abst}(\omega_1) = \mathrm{abst}(\rho) \mid \mathrm{res}(\omega_1, f(\rho)) = \mathrm{res}(\rho)] \\ &= A^{f(\rho)}_{\mathrm{res}(\rho)} \cdot \mathsf{Pr}_{\omega_1}[\mathrm{abst}(\omega_1) = \mathrm{abst}(\rho) \mid \mathrm{res}(\omega_1, f(\rho)) = \mathrm{res}(\rho)]\end{aligned}$$

and similarly

$$\mathsf{Pr}_{\omega_1}[\tilde{\varphi}_\rho(\omega_1)] = A^{i}_{\mathrm{res}(\rho)} \cdot \mathsf{Pr}_{\omega_1}[\mathrm{abst}(\omega_1) = \mathrm{abst}(\rho) \mid \mathrm{res}(\omega_1, i) = \mathrm{res}(\rho)].$$

Furthermore, we have

$$\begin{aligned}\mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = \text{abst}(\rho) \mid \text{res}(\omega_1, f(\rho)) = \text{res}(\rho)] =&\\ &= \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = \text{abst}(\rho) \mid r_0(\omega_1) = r_0(\rho)]\\ &= \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = \text{abst}(\rho) \mid \text{res}(\omega_1, i) = \text{res}(\rho)]\end{aligned}$$

as the set of abstaining voters depends only on the number of abstaining voters. Together with Lemma 1, we immediately obtain for all ω_1^ρ with $\varphi_\rho(\omega_1^\rho)$:

$$\begin{aligned}\mathsf{Pr}[T \mapsto \rho] - \mathsf{Pr}[\tilde{T} \mapsto \rho] =& \big(A^{f(\rho)}_{\text{res}(\rho)} - A^{i}_{\text{res}(\rho)}\big) \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^\rho, \omega_2) \mapsto \rho] \\ &\cdot \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = \text{abst}(\rho) \mid r_0(\omega_1) = r_0(\rho)].\end{aligned} \tag{4}$$

Note that if there does not exist $\tilde{\omega}_1^\rho$ such that $\tilde{\varphi}_\rho(\tilde{\omega}_1^\rho)$, then $A^{i}_{\text{res}(\rho)} = 0$ and $\mathsf{Pr}[\tilde{T} \mapsto \rho] = 0$.

As shown in Appendix B.3, (4) can now be used to prove that $\mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1] \leq \delta$, which concludes the proof of Theorem 1.

5 Related Work

As mentioned in the introduction, only very few voting protocols used in practice have been analyzed rigorously in cryptographic models w.r.t. coercion-resistance. The lack of suitable cryptographic definitions of coercion-resistance has been a major obstacle, which also becomes clear from our case study: As discussed in Section 4.1, our definition of coercion-resistance proposed recently [13] is in fact the only definition suitable for analyzing Scantegrity II. We refer the reader to [13] for a detailed discussion and comparison of definitions of coercion-resistance, and on the voting protocols these definitions have been applied to. In what follows, we discuss the analysis of voting protocols that have actually been used in practice w.r.t. coercion-resistance.

Juels et al. [10] sketched a proof of coercion-resistance of their voting protocol based on their definition of coercion-resistance. A generalized version of the protocol by Juels et al. was later implemented in the Civitas system [5].

In [13], we applied our definition of coercion-resistance to ThreeBallot [17] and Bingo Voting [2], showing that Bingo Voting provides the ideal level of coercion-resistance, but ThreeBallot does not. In [13], we also pointed out that other cryptographic definitions of coercion-resistance are not suitable for the analysis of ThreeBallot and Bingo Voting.

Several definitions of coercion-resistance were proposed in symbolic, Dolev-Yao-style models (see, e.g., [7, 12]). These models (and definitions) are less accurate than cryptographic models since an abstract view on cryptography is taken. As a result, analysis in these models is simpler, but provides weaker security guarantees. Several voting protocols have been analyzed based on symbolic definitions, with the prominent Civitas system [5] analyzed in [12].

6 Conclusion and Future Work

In this paper, we have shown that Scantegrity II provides an optimal level of coercion-resistance, i.e., the same level of coercion-resistance as an ideal voting protocol, under the (necessary) assumption that the workstation used in the protocol is honest. Since we assume that the coercer can see the receipts of all voters, and hence, he can see whether or not a voter voted, Scantegrity II is not resistant to forced abstentation attacks.

Besides coercion-resistance, Scantegrity II is also designed to provide verifiability. We leave it to future work to analyze Scantegrity II w.r.t. this property. It should be possible to use our recently proposed definition of verifiability [14] for this purpose. In [14], we also provide a definition of accountability, which would be interesting to apply to Scantegrity II too.

References

1. J. C. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing (STOC 1994)*, pages 544–553. ACM Press, 1994.
2. J.-M. Bohli, J. Müller-Quade, and S. Röhrich. Bingo Voting: Secure and Coercion-Free Voting Using a Trusted Random Number Generator. In A. Alkassar and M. Volkamer, editors, *E-Voting and Identity (VOTE-ID 2007)*, volume 4896 of *Lecture Notes in Computer Science*, pages 111–124. Springer, 2007.
3. D. Chaum. Secret-Ballot Receipts: True Voter-Verifiable Elections. *IEEE Security & Privacy*, 2(1):38–47, 2004.
4. D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. L. Rivest, P. Y. A. Ryan, E. Shen, and A. T. Sherman. Scantegrity II: End-to-End Verifiability for Optical Scan Election Systems using Invisible Ink Confirmation Codes. In *USENIX/ACCURATE Electronic Voting Technology (EVT 2008)*. USENIX Association, 2008. See also `http://www.scantegrity.org/elections.php`.
5. M. R. Clarkson, S. Chong, and A. C. Myers. Civitas: Toward a Secure Voting System. In *2008 IEEE Symposium on Security and Privacy (S&P 2008)*, pages 354–368. IEEE Computer Society, 2008.
6. Josh D. Cohen and Michael J. Fischer. A Robust and Verifiable Cryptographically Secure Election Scheme (Extended Abstract). In *26th Annual Symposium on Foundations of Computer Science (FOCS 1985)*, pages 372–382. IEEE, 1985.
7. S. Delaune, S. Kremer, and M.D. Ryan. Coercion-Resistance and Receipt-Freeness in Electronic Voting. In *Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW'06)*, pages 28–39. IEEE Computer Society Press, 2006.
8. Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A Practical Secret Voting Scheme for Large Scale Elections. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology (AUSCRYPT 1992)*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992.
9. O. Goldreich. *Foundations of Cryptography*, volume 1. Cambridge Press, 2001.
10. A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant Electronic Elections. In *Proceedings of Workshop on Privacy in the Eletronic Society (WPES 2005)*, pages 61–70. ACM Press, 2005.

11. R. Küsters. Simulation-Based Security with Inexhaustible Interactive Turing Machines. In *Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW-19 2006)*, pages 309–320. IEEE Computer Society, 2006.
12. R. Küsters and T. Truderung. An Epistemic Approach to Coercion-Resistance for Electronic Voting Protocols. In *2009 IEEE Symposium on Security and Privacy (S&P 2009)*, pages 251–266. IEEE Computer Society, 2009.
13. Ralf Küsters, Tomasz Truderung, and Andreas Vogt. A Game-based Definition of Coercion-Resistance and its Applications. In *23th IEEE Computer Security Foundations Symposium, CSF 2010*, pages 122–136. IEEE Computer Society, 2010.
14. Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: Definition and Relationship to Verifiability. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS 2010)*. ACM, 2010. To appear. Also available as Technical Report 2010/236, Cryptology ePrint Archive, 2010. http://eprint.iacr.org/2010/236/.
15. T. Moran and M. Naor. Receipt-Free Universally-Verifiable Voting With Everlasting Privacy. In C. Dwork, editor, *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Proceedings*, volume 4117 of *Lecture Notes in Computer Science*, pages 373–392. Springer, 2006.
16. C. A. Neff. Practical High Certainty Intent Verification for Encrypted Votes. `http://www.votehere.com/old/vhti/documentation/vsv-2.0.3638.pdf`.
17. R. L. Rivest and W. D. Smith. Three Voting Protocols: ThreeBallot, VAV and Twin. In *USENIX/ACCURATE Electronic Voting Technology (EVT 2007)*, 2007.
18. P. Y. A. Ryan. A variant of the Chaum voter-verifiable scheme. In *Water, Innovation, Technology & Sustainability (WITS 2005)*, pages 81–88, 2005.
19. V. Teague, K. Ramchen, and L. Naish. Coercion-Resistant tallying for STV voting. In *IAVoSS Workshop On Trustworthy Elections (WOTE 2008)*, 2008.

A Modeling of Scantegrity II

As mentioned, our formal modeling of the Scantegrity II system is based on a quite standard computational model, similar to models for simulation-based security [11], in which *inexhaustible interactive Turing machines (IITMs)* communicate via tapes. In this model, only one IITM is active at a time. Such a machine may perform computations polynomially bounded by the security parameter and the input of its tapes. It may send a message to another IITM which is then triggered. The systems of IITMs we consider will always run in polynomial time in the length of the security parameter (see below).

There is one special IITM, called the master, which is triggered first. This machine is also triggered if, at some point, no message is sent. In our modeling, this machine will play the role of a *scheduler* which makes sure that every party gets a chance to perform the necessary actions in every protocol phase.

We model the Scantegrity II system as an election system $S = \mathsf{P}_{\mathsf{Sct}}(k, m, n, \boldsymbol{p})$ in the sense of Section 2.2 in the following way.

Participants. The set Σ of protocol participants is defined as follows:

(a) **The voters** $\mathsf{v}_0, \ldots, \mathsf{v}_m$. The voters $\mathsf{v}_1, \ldots, \mathsf{v}_n$ are supposed to be honest, i.e., these voters run their programs as specified in the protocol. The voter

v_0 is **the coerced voter**. The voters $\mathsf{v}_{n+1}, \ldots, \mathsf{v}_m$ are dishonest and will be subsumed by the coercer.

(b) **Voting authorities**: One group of authorities are the clerks $cl_1, \ldots, cl_r$. Then, there is the main component, the workstation WSt (which also includes the pseudo random number generator PRNG and the scanner)). Moreover, there are auditors $\mathsf{A}_1, \ldots, \mathsf{A}_t$. We assume that the workstation (including the PRNG and scanner) is honest. We also assume that at least one of the clerks, say cl_1, is honest (all others will be subsumed by the coercer); all auditors may be dishonest, and in fact, will be subsumed by the coercer.

(c) **The coercer** c.

(d) **The scheduler** (the master) Sch.

By Σ_h we denote the set $\{\mathsf{v}_0, \ldots, \mathsf{v}_n, cl_1, \mathsf{WSt}, \mathsf{Sch}\}$ of honest participants plus the coerced voter. By Σ_d we denote the set $\Sigma \setminus \Sigma_h$ of dishonest participants.

Channels. The set of channels (tapes) consists of the channels ch_b^a, for every $a, b \in \Sigma$. The channel ch_b^a is an output channel of a and an input channel of b. Therefore a and b can communicate using ch_b^a and ch_a^b.[2]

In the following, we complete the definition of $S = \mathsf{P}_{\mathsf{Sct}}(k, m, n, \boldsymbol{p})$ by providing the description of programs (IITMs) of the honest participants (honest voters and authorities), the coerced voter, the coercer, and the scheduler. Note that, since we do not assume that the auditors are honest, these parties will be subsumed by the coercer (which technically means that the coercer will have access to the channels of these parties). Hence, we do not need to provide a description of the programs run by the auditors.

The coercer: The set C_S of programs of the coercer contains all the IITMs with input channels (tapes) of the form ch_b^a and output channels of the form ch_a^b, for $a \in \Sigma_h$ and $b \in \Sigma_d$.

The coerced voter: The set V_S of programs of the coerced voter v_0 contains all the IITMs with input channels of the form $\mathsf{ch}_{\mathsf{v}_0}^a$ and output channels of the form $\mathsf{ch}_a^{\mathsf{v}_0}$, for $a \in \Sigma$.

[2] Technically, following the terminology of [11], we assume that the channels of the form ch_b^a, for $a \in \Sigma_d$ and $b \in \Sigma_h$, i.e., the channels from the dishonest parties to the honest parties, are modeled by so-called *enriching tapes*. All the remaining channels are modeled by so-called *consuming tapes*. This architecture guarantees that polynomially bounded IITMs plugged together form a polynomially bounded system. Also, the power of the coercer is not restricted by this choice, due to the universal quantification of the programs of the coercer in our definition of coercion resistance. Moreover, honest parties can perform any polynomial-time computation and produce any output in reaction to all inputs.

The scheduler: The role of the scheduler is to fairly schedule the protocol, giving every participant, including the coercer and the coerced voter, a chance to perform the necessary actions in every stage of the voting procedure.

As we noted before, the scheduler, as the master, is triggered at the beginning of the protocol execution and, then, whenever no message is sent. Because, every IITM in the system is polynomially bounded, after the scheduler gives the control to some other machine, it must eventually happen that no message is sent and the scheduler is triggered again. Therefore, the scheduler is able to perform the steps described below, whatever the remaining IITMs in the system do.

1. *Initialization*: The scheduler triggers the workstation (allowing it to perform the initialization procedure).
2. *Voting*: The scheduler triggers every voter (allowing her to vote).
3. *Receipt checking*: The scheduler triggers every voter (allowing her to check her receipt).
4. *Tallying*: The scheduler triggers the workstation (allowing it to perform the tallying phase).

Furthermore, before the first step, after the last one, and between each two successive ones, the scheduler triggers the coercer and the coerced voter (by this the coercer can perform some steps, perhaps involving communication with the coerced voter, before the election process starts and, then, between the protocol phases).

An honest clerk: The program of the honest clerk cl_1, when requested by the workstation, generates a random number (its share of the seed) and sends it back to the workstation (using channel $\mathsf{ch}^{cl_1}_{\mathsf{WSt}}$). This is the only action this program performs.

Honest voters: The program of an honest voter performs the following voting procedure, when triggered by the scheduler for the first time (which happens in the voting phase):

It uses the probability distribution $\boldsymbol{p}$ to make its choice (including abstention). If the choice is to abstain from voting, the program halts (it will ignore all successive messages). Otherwise, the program decides (using a random bit) whether it demands one or two ballots, sends the appropriate request to the workstation, and obtains a message containing a description of one or two ballots. Then, if two ballots were demanded, the (program of the) voter decides which one is to be audited and sends this decision to the workstation (which is supposed to open the corresponding commitments); the second ballot will be used to vote. Next, the program sends to the workstation its chosen candidate. The workstation replies revealing the code on the ballot next to the chosen candidate (recall that the workstation is assumed to be honest).

Then, when triggered in the receipt checking phase, the program of an honest voter reveals her receipt (that is the serial number and the revealed code on the used ballot), by sending it to the coercer.

The workstation: The workstation, besides performing its protocol steps as described below, maintains a bulletin board, which we model in the following way: Whenever the workstation is requested by some party (by sending some fixed message), it replies with the current content of the bulletin board.

Furthermore, the program (IITM) of the workstation performs the following steps. First, when triggered by the scheduler for the first time (in the initialization phase), it requests to receive the (partial) seeds from all (honest and dishonest) clerks. It then takes the XOR of these seeds and gives the resulting seed to the PRNG, which returns a pseudo-random bit string. Based on this pseudo-random bit string, the workstation then computes all the values (tables), as described in Section 3.1, and publishes these values on the bulletin board, i.e., outputs them on request.

In the voting phase, when requested by a voter v_i, it sends to v_i a description of one or two ballots (as requested). When v_i requests one of the ballots to be audited, the (program of the) workstation opens all the required commitments, as specified in Section 3.1; formally, the workstation adds the open messages to the content of the bulletin board, which, again, is provided to participants on request. Then, after v_i sends to the workstation her choice, the workstation sends back to v_i the appropriate code. The workstation stores the choices made by the voters along with the serial numbers and the revealed codes.

When the workstation is triggered by the scheduler for the second time, it performs the steps of the tallying phase, as described in Section 3.1. Since the auditors are assumed to be dishonest, i.e,. subsumed by the coercer, the workstation requests the coercer to send the random coins for the auditing step.

B Details of the Proof of Theorem 1

In this section, we provide further details of the proof of Theorem 1.

B.1 Replacing the Pseuo-Random String by a Real Random String

We first note that the seed given to the PRNG is chosen uniformly at random, since, by assumption, one of the clerks is assumed to be honest. By the definition of PRNGs (see, e.g., [9]), the distribution of (pseudo-random) bit strings produced by a PRNG, given a random bit string as input, is computationally indistinguishable from real random bit strings.

Now let $\tilde{T} = (\tilde{v} \parallel c \parallel \mathsf{e}_S)$ and $T = (\mathsf{dum} \parallel c \parallel \mathsf{e}_S)$, where in e_S, a real random bit strings is used instead of the pseudo-random bit string produced by the PRNG. Let $\tilde{T}' = (\tilde{v} \parallel c \parallel \mathsf{e}'_S)$ and $T' = (\mathsf{dum} \parallel c \parallel \mathsf{e}'_S)$, where in e'_S, the PRNG is used (as specified in Section 3.1). By definition of a PRNG, we get that $|\mathsf{Pr}[T' \mapsto 1] - \mathsf{Pr}[T \mapsto 1]|$ and $|\mathsf{Pr}[\tilde{T}' \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1]|$ are negligible in the security parameter as otherwise the output of the coercer could be used to distinguish the PRNG from a real random bit string. Hence, if we can show that the difference between $\mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1]$ is negligible, it follows that the difference $\mathsf{Pr}[T' \mapsto 1] - \mathsf{Pr}[\tilde{T}' \mapsto 1]$ is negligible as well. So, in the remaining proof of Theorem 1, it suffices to assume that the workstation is given a real random bit string.

B.2 The Cryptographic Part: Proof of Lemma 1

The core of Lemma 1 is stated in the following lemma.

Lemma 2. *Let ρ be an arbitrary view such that $f(\rho)$ is defined. Let $\omega_1, \omega_1', \omega_1'', \omega_1'''$ be arbitrary, fixed elements of Ω_1 with $\varphi_\rho(\omega_1), \varphi_\rho(\omega_1')$, $\tilde{\varphi}_\rho(\omega_1'')$, and $\tilde{\varphi}_\rho(\omega_1''')$. Then the sets*

$$\begin{aligned} A &= \{\omega_2 : T(\omega_1, \omega_2) \mapsto \rho\}, & C &= \{\omega_2 : \tilde{T}(\omega_1'', \omega_2) \mapsto \rho\}, \\ B &= \{\omega_2 : T(\omega_1', \omega_2) \mapsto \rho\}, & D &= \{\omega_2 : \tilde{T}(\omega_1''', \omega_2) \mapsto \rho\}. \end{aligned}$$

have the same cardinality, and hence, $\mu_2(A) = \mu_2(B) = \mu_2(C) = \mu_2(D)$.

To prove this lemma, we use Lemma 3. To state Lemma 3, we use the following notation. By $\tilde{T}_j$ we denote the system $(\tilde{v}_j \parallel c \parallel \mathsf{e}_S)$, where $\tilde{v}_j$ is defined like $\tilde{v}$ but votes for j instead of i. So we have $\tilde{v} = \tilde{v}_i$ and $\tilde{T} = \tilde{T}_i$. Moreover, for each view ρ of the coercer, for which $f(\rho)$ is defined, we clearly have: $T(\omega_1, \omega_2) \mapsto \rho$ iff $\tilde{T}_{f(\rho)}(\omega_1, \omega_2) \mapsto \rho$. A permutation σ on a tuple $(v_0, \dots, v_n) \in \{0, 1, \dots, k\}^{n+1}$ is a permutation on the set of indices $\{0, \dots, n\}$. We write $\sigma(v_0, \dots, v_n)$ for the tuple $(v_{\sigma(0)}, \dots, v_{\sigma(n)})$. For simplicity of notation, we sometimes write $\sigma(v_i)$ instead of $v_{\sigma(i)}$. We say that σ does not change the abstaining votes of $(v_0, \dots, v_n)$ if $\sigma(j) = j$ for every $j \in \{0, \dots, k\}$ with $v_j = 0$. For $j \in \{1, \dots, k\}$ and $\omega_1 \in \Omega_1 (= \{0, 1, \dots, k\}^n)$, we consider (j, ω_1) to be an $(n+1)$-tuple over $\{0, 1, \dots, k\}$. If σ is a permutation on (j, ω_1), we may apply σ to ω_1, written $\sigma(\omega_1)$, with the obvious meaning. With this and the above conventions, we have that $\sigma(j, \omega_1) = (\sigma(j), \sigma(\omega_1))$.

Lemma 3. *For every $j \in \{1, \dots, k\}$, every $\omega_1 \in \Omega_1$ and every permutation σ^0 on (j, ω_1) that does not change the abstaining votes, there is a bijective function $h = h^{j, \omega_1, \sigma^0}$ from Ω_2 to Ω_2 such that for all ω_2 we have that $\tilde{T}_j(\omega_1, \omega_2)$ yields the same view as $\tilde{T}_{\sigma^0(j)}(\sigma^0(\omega_1), h(\omega_2))$.*

We postpone the proof of this lemma to the end of this section. Now, Lemma 2 follows directly from Lemma 3: Given the assumptions of Lemma 2, there are permutations σ_1^0, σ_2^0, and σ_3^0 such that $(f(\rho), \omega_1) = \sigma_1^0(f(\rho), \omega_1') = \sigma_2^0(i, \omega_1'') = \sigma_3^0(i, \omega_1''')$. Moreover, $T(\omega_1, \omega_2) \mapsto \rho$ iff $\tilde{T}_{f(\rho)}(\omega_1, \omega_2) \mapsto \rho$ and $\tilde{T}(\omega_1, \omega_2) \mapsto \rho$ iff $\tilde{T}_i(\omega_1, \omega_2) \mapsto \rho$. From this and Lemma 3 we obtain that the functions $h^{f(\rho), \omega_1, (\sigma_1^0)^{-1}}$, $h^{f(\rho), \omega_1, (\sigma_2^0)^{-1}}$, and $h^{f(\rho), \omega_1, (\sigma_3^0)^{-1}}$, are bijections between A and B, A and C, and A and D, respectively.

Now with Lemma 2 we can easily complete the proof of Lemma 1:

$$
\begin{aligned}
\mathsf{Pr}[T \mapsto \rho] &= \mathsf{Pr}_{\omega_1,\omega_2}[\varphi_\rho(\omega_1), T(\omega_1,\omega_2) \mapsto \rho] \\
&= \sum_{\omega_1' : \varphi_\rho(\omega_1')} \mathsf{Pr}_{\omega_1,\omega_2}[\omega_1 = \omega_1', T(\omega_1',\omega_2) \mapsto \rho] \\
&= \sum_{\omega_1' : \varphi_\rho(\omega_1')} \mu_1(\omega_1') \cdot \mathsf{Pr}_{\omega_1,\omega_2}[T(\omega_1',\omega_2) \mapsto \rho \mid \omega_1 = \omega_1'] \\
&= \sum_{\omega_1' : \varphi_\rho(\omega_1)} \mu_1(\omega_1') \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1',\omega_2) \mapsto \rho] \\
&= \sum_{\omega_1' : \varphi_\rho(\omega_1)} \mu_1(\omega_1') \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^\rho,\omega_2) \mapsto \rho] \\
&= \mathsf{Pr}_{\omega_1}\left[\varphi_\rho(\omega_1)\right] \cdot \mathsf{Pr}_{\omega_2}\left[T(\omega_1^\rho,\omega_2) \mapsto \rho\right].
\end{aligned}
$$

This proves (1). The proof for (2) is analogous. Statement (3) follows immediately from Lemma 2.

Proof of Lemma 3. To prove Lemma 3, we first introduce notation for the components (cryptographic operations, random numbers, etc.) of the Scantegrity II protocol.

The cryptographic components. We first describe in detail the structure of the sequence $\omega_2 \in \Omega_2$ of random coins. In the following, by $\mathsf{comm}(a, r)$ we denote the commitment on a with randomness r.

(a) α — the random coins of the coercer.
(b) c_i^j for $i \in \{1, \dots, s\}$ and $j \in \{1, \dots, k\}$ — the codes.
(c) π_i for $i \in \{1, \dots, s\}$ — the permutations used to create the Q-table from the P-table (row i of P is permutated by π_i).
(d) π, — a permutation of $1, \dots, s \cdot k$ representing the Q-pointers of table R.
(e) π', — a permutation of $1, \dots, s \cdot k$ representing the S-pointers of table R, where we assume that π' is consistent with π and π_i, i.e. a every element of the P-table is mapped to the right column in the S-table.
(f) r_i^j for $i \in \{1, \dots, s\}$ and $j \in \{1, \dots, k\}$ — the random numbers used for the commitments of the entries in the Q-table, $cc_i^j = \mathsf{comm}(\pi_i(c_i^j), r_i^j)$.
(g) R_t^1, R_t^2 for $t \in \{1, \dots, s \cdot k\}$ — the random numbers used for the commitments of the entries in the R-table, $cR_t^1 = \mathsf{comm}(\pi(t), R_t^1)$ and $cR_t^2 = \mathsf{comm}(\pi'(t), R_t^2)$.
(h) Random values S_u for $u = 1, \dots, n$ — determining whether the honest voter v_u takes two ballots and which of the ballots to audit.

A view ρ of the coercer, depending on ω_2 and the choices $v_0, \dots, v_n$ taken by the voters, consists of the following parts, where with s_i for $i \in \{1, \dots, s\}$ we denote the random bits that the coercer determines on behalf of the auditors, see Section A:

(B1) α — random coins of the coercer.

(B2) The commitments $cc_i^j = \mathsf{comm}((c_i^{\pi_i(j)}), r_i^j)$, $cR_t^1 = \mathsf{comm}(\pi(t), R_t^1)$, and $cR_t^2 = \mathsf{comm}(\pi'(t), R_t^2)$.

(B3) $c_u^{v_u}$ — the code of the u-th voter, for every non-abstaining voter u (voting for candidate v_u).

(B4) The content of all the ballots that are audited and the codes revealed by the dishonest voters. In case the dishonest voters reveal more than one code (and hence, make the ballot invalid), are also revealed to the coercer.

(B5) The opened commitments in the Q-table.

(B6) The opened commitments of the R-table corresponding to the challenges s_i.

(B7) The flagged entries in the R- and S-table.

More precisely, the view might consist only of a subset of these parts, as the coercer possibly does not ask the workstation for some of these values, or he might even refuse to send the seeds on behalf of the auditors, see Section A.

Because every permutation is the finite composition of permutations that switch only two successive positions, it suffices to consider the case where σ flips the positions l and $l+1$; the rest follows from composing permutations and bijections. Let $\tilde{v}_0, \ldots, \tilde{v}_n$ be such that

$$\sigma(v_0, \ldots, v_n) = (\tilde{v}_0, \ldots, \tilde{v}_n) = (v_0, \ldots, v_{l+1}, v_l, \ldots, v_n.)$$

Further, we assume that $v_l = y \neq z = v_{l+1}$, as the case that $\sigma(v_0, \ldots, v_n) = (v_0, \ldots, v_n)$ is trivial. Recall that, by assumption, we have that $y, z \neq 0$.

Let ω_2 be any element of Ω_2 and let α, c_i^j, π_i, π, π', r_i^j, R_t^1, R_t^2, and S_u be the parts of ω_2 defined as above. Here, i ranges over $1, \ldots, s$, j over $1, \ldots, k$, t over $1, \ldots, s \cdot k$, and u over $1, \ldots, n$. We will denote the corresponding parts of $h(\omega_2)$ by $\tilde{\alpha}$, $\tilde{c}_i^j$, and so on. We define $h(\omega_2)$ as follows:

- $\tilde{\alpha} = \alpha$. As one can see, (B1) remains unchanged.
- $\tilde{c}_i^j$ are defined like c_i^j, except for that c_l^y and c_l^z as well as c_{l+1}^z and c_{l+1}^y are swapped. By that, (B3) remains unchanged.
- $\tilde{\pi}_i$ is defined as π_i, except for $\tilde{\pi}_l$ and $\tilde{\pi}_{l+1}$. These two permutations differ from π_l and π_{l+1} in that the positions of c_l^y and c_l^z and of c_{l+1}^z and c_{l+1}^y are swapped.
- $\tilde{r}_i^j$ are defined like r_i^j. By that we get that (B2) remains unchanged.
- The rest of ω_2 remains unchanged. By that, (B4), (B5), (B6), and (B7) remain unchanged.

This concludes the description of $h(\omega_2)$. As we have noted, all the parts (B1)–(B7) of the views in both cases—for ω_2 and $h(\omega_2)$—are exactly the same. As h just flips two times two codes and changes two corresponding permutations, h is a bijection from Ω_2 to Ω_2.

B.3 The Combinatorial Part: Reduction to the Ideal Case

In this section, we complete the reduction of the combinatorial part of the proof to the ideal case. More precisely, we use (4) to show that

$$\mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1] \leq \delta^i_{min}(n, k, \boldsymbol{p}) = \delta.$$

Let M be the set of views that are accepted by the program c of the coercer, i.e., for which the coercer outputs 1. In what follows, let j range over the set of candidate names $\{1, \ldots, k\}$, $\boldsymbol{r} = (r_0, \ldots, r_k)$ over all the pure results and S over all subsets of $\{1, \ldots, n\}$. Let $M_j^{\boldsymbol{r},S} = \{\rho \in M : f(\rho) = j,\ \text{abst}(\rho) = S \text{ and } \text{res}(\rho) = \boldsymbol{r}\}$. In the following, we can assume without loss of generality that M contains only views ρ such that $\mathsf{Pr}[T \mapsto \rho] > 0$ (this is because, by removing from M views that fail to satisfy this condition, we only make the expression $\mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1]$ bigger). Therefore, for all j, $\boldsymbol{r}$, and S such that $M_j^{\boldsymbol{r},S}$ is non-empty, there exists $\omega_1^{j,\boldsymbol{r},S}$ such that $\text{res}(\omega_1^{j,\boldsymbol{r},S}, j) = \boldsymbol{r}$ and $\text{abst}(\omega_1^{j,\boldsymbol{r},S}) = S$. Clearly, we have $\varphi_\rho(\omega_1^{j,\boldsymbol{r},S})$ for all $\rho \in M_j^{\boldsymbol{r},S}$. We have

$$\begin{aligned}
\Phi &= \mathsf{Pr}[T \mapsto 1] - \mathsf{Pr}[\tilde{T} \mapsto 1] = \mathsf{Pr}[T \mapsto M] - \mathsf{Pr}[\tilde{T} \mapsto M] \\
&= \sum_j \sum_{\boldsymbol{r}} \sum_S \sum_{\rho \in M_j^{\boldsymbol{r},S}} \big(\mathsf{Pr}[T \mapsto \rho] - \mathsf{Pr}[\tilde{T} \mapsto \rho]\big) \\
&= \sum_j \sum_{\boldsymbol{r}} \sum_S \sum_{\rho \in M_j^{\boldsymbol{r},S}} (A^j_{\boldsymbol{r}} - A^i_{\boldsymbol{r}}) \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho] \cdot \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = S | r_0(\omega_1) = r_0] \\
&= \sum_j \sum_{\boldsymbol{r}} (A^j_r - A^i_r) \sum_S \sum_{\rho \in M_j^{\boldsymbol{r},S}} \mathsf{Pr}_{\omega_2}[T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho] \cdot \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = S | r_0(\omega_1) = r_0].
\end{aligned}$$

For the third equation we used that $\mathsf{Pr}[T \mapsto \rho] - \mathsf{Pr}[\tilde{T} \mapsto \rho] = 0$ if $f(\rho)$ is not defined. Let $M^*_{i,j} = \{\boldsymbol{r} : A^j_{\boldsymbol{r}} \geq A^i_{\boldsymbol{r}}\}$. Then, we obtain

$$\Phi \leq \sum_j \sum_{\boldsymbol{r} \in M^*_{i,j}} (A^j_{\boldsymbol{r}} - A^i_{\boldsymbol{r}}) \sum_S \sum_{\rho \in M_j^{\boldsymbol{r},S}} \mathsf{Pr}_{\omega_2}[T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho] \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = S \mid r_0(\omega_1) = r_0].$$

Next, we use that, by the definition of $M_j^{\boldsymbol{r},S}$, for $\rho \in M_j^{\boldsymbol{r},S}$ we have $f(\rho) = j$ and, because $f(\rho)$ depends only on ω_2, $T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho$ implies $f(\omega_2) = j$. With this, we obtain

$$\mathsf{Pr}_{\omega_2}[T(\omega_1^\rho, \omega_2) \mapsto \rho] = \mathsf{Pr}_{\omega_2}[f(\omega_2) = j] \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho \mid f(\omega_2) = j]$$

for $\rho \in M_j^{\boldsymbol{r},S}$. Now, we can conclude

$$\begin{aligned}
\Phi &\le \sum_j \sum_{\boldsymbol{r}\in M_{i,j}^*} (A_{\boldsymbol{r}}^j - A_{\boldsymbol{r}}^i) \sum_S \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = S | r_0(\omega_1) = r_0] \\
&\qquad \sum_{\rho \in M_j^{\boldsymbol{r},S}} \mathsf{Pr}_{\omega_2}[f(\omega_2) = j] \cdot \mathsf{Pr}_{\omega_2}[T(\omega_1^{j,\boldsymbol{r},S}, \omega_2) \mapsto \rho \mid f(\omega_2) = j] \\
&\le \sum_j \mathsf{Pr}_{\omega_2}[f(\omega_2) = j] \sum_{r\in M_{i,j}^*} (A_r^j - A_r^i) \sum_S \mathsf{Pr}_{\omega_1}[\text{abst}(\omega_1) = S | r_0(\omega_1) = r_0] \\
&\le \sum_j \mathsf{Pr}_{\omega_2}[f(\omega_2) = j] \sum_{\boldsymbol{r}\in M_{i,j}^*} (A_{\boldsymbol{r}}^j - A_{\boldsymbol{r}}^i) \\
&\le \sum_j \mathsf{Pr}_{\omega_2}[f(\omega_2) = j] \cdot \delta_{min}^i(n, k, \boldsymbol{p}) \ \le \delta_{min}^i(n, k, \boldsymbol{p}) = \delta.
\end{aligned}$$

This concludes the proof of Theorem 1.