Towards Accountability in CRS Generation

Prabhanjan Ananth[1], Gilad Asharov[2], Hila Dahari[3], and Vipul Goyal[4]

[1] University of California, Santa Barbara,
prabhanjan@cs.ucsb.edu
[2] Bar-Ilan University, Israel
gilad.asharov@biu.ac.il
[3] Weizmann Institute, Israel
hila.dahari@weizmann.ac.il
[4] NTT Research and Carnegie Mellon University
vipul@cmu.edu

Abstract. It is well known that several cryptographic primitives cannot be achieved without a common reference string (CRS). Those include, for instance, non-interactive zero-knowledge for NP, or maliciously secure computation in fewer than four rounds. The security of those primitives heavily relies upon on the assumption that the trusted authority, who generates the CRS, does not misuse the randomness used in the CRS generation. However, we argue that there is no such thing as an unconditionally trusted authority and every authority must be held accountable for any trust to be well-founded. Indeed, a malicious authority can, for instance, recover private inputs of honest parties given transcripts of the protocols executed with respect to the CRS it has generated.
While eliminating trust in the trusted authority may not be entirely feasible, can we at least move towards achieving some notion of accountability? We propose a new notion in which, if the CRS authority releases the private inputs of protocol executions to others, we can then provide a publicly-verifiable proof that certifies that the authority misbehaved. We study the feasibility of this notion in the context of non-interactive zero knowledge and two-round secure two-party computation.

1 Introduction

Very broadly, cryptography can be seen as having two parallel lines of research: one where the parties don't trust anyone but themselves, and another where security relies on some kind of trust assumption. Most notably, many works have relied on the common reference string (CRS) model where a trusted party chooses and publishes a public string. The advantage of relying on a CRS depends upon the setting. For example, for ZK it is known that while in the CRS model, a non-interaction solution can be achieved [9, 17] one needs at least 3 rounds in the plain model [22]. For MPC, two rounds are sufficient in the CRS model [8, 21, 29] while the best known constructions in the plain model require at least 4-rounds [1, 4, 10, 15, 27, 28]. Furthermore, UC security is known to be impossible to achieve in the plain model [11, 12] while this impossibility can be bypassed

in the CRS model [11,13]. Thus, while one might prefer to obtain constructions in the plain model, it seems unlikely that the CRS model will be abandoned anytime soon.

Do There Really Exist Trusted Parties? We argue that in real life, there is no such thing as an unconditional trusted party. We argue that the only reason we trust a party is because the cost of cheating (if caught) for that party would be much higher than the potential gains obtained by cheating successfully. Indeed this applies everywhere in our society. We are more comfortable trusting a large bank with our personal information compared to a small individual lender only because a large bank will pay a much higher cost (loss of reputation and potential future business) if it behaves maliciously. However if the cost to the large bank was zero, the reasons for placing this trust would be unfounded. There are multiple examples where even large entities systematically participated in an activity which would be generally unacceptable only because they thought the activity would not become public knowledge (e.g., Facebook selling data to Cambridge Analytica, or Wells Fargo opening accounts without customer knowledge).

Compared to real life, in cryptography, life is largely black and white: dishonest parties can be arbitrarily bad and trusted parties are unconditionally trusted. For example, the party generating a CRS (referred to as the CRS authority from hereon) in a NIZK system can potentially even recover your witness entirely and sell it for profit. Similarly, in MPC, the CRS authority may recover your input and pass it on to another party. Even if you detect that the authority is doing that, it's not clear how to prove it in a court of law and seek damages. You can publicly blame the authority for doing that. But this is then indistinguishable from a malicious party blaming an honest authority.

These concerns have motivated the study of weaker notions such as ZAPs [16] and super-polynomial simulation security [30]. Groth and Ostrovsky studied the so called multi-string model [25] where multiple authorities publish common reference strings such that a majority of them are guaranteed to be honest. Goyal and Katz [24], and later Garg et. al [20] studied UC security with an unreliable CRS if the CRS turns out to be malicious, some other setup or an honest majority can come to the rescue. Bellare et. al [7] studied NIZKs with an untrusted CRS where even if the CRS is malicious, some weaker security properties still hold.

In this work, we focus on a single CRS while providing some notion of accountability towards the CRS generation authority. Our direction is orthogonal to many of the works mentioned and, to our knowledge, largely unexplored.

Towards Accountability in CRS Generation: While eliminating trust in the CRS authority entirely may not be feasible, can we at least move towards achieving some notion of accountability? As an example, suppose you find out that the CRS authority decrypted your input used in an execution of MPC protocol and sold to another party for profit. Can you obtain a cryptographic proof of this fact? Can you convince others that such an incident has happened? Indeed there are limits on what can and cannot be achieved. For example, if

the authority sells your input and you never find out, it's unclear if something can be done. But if the decrypted input indeed falls into your hand (e.g., the person buying the input from the authority was your own agent), you know for sure that the authority is dishonest (although you may not be able to prove it to others).

In this work, we study if there is *any* meaningful notion of accountability that can be achieved with respect to a CRS authority. We focus specifically on the case of NIZK, and two-round MPC which are both known to be impossible in the plain model and yet achievable with a CRS. Our work runs into several novel technical challenges as we discuss later. We note that our study is far from complete and leaves open various intriguing questions.

1.1 Our Results

In this work, we propose novel notions of accountability in the context of two party secure computation protocols and NIZKs. We also accompany these definitions with constructions realizing these notions.

Secure Two-Party Computation (2PC). Our definition of malicious authority security first requires the same security guarantees as in regular secure computation, that is, if the CRS was honestly generated, then the protocol achieves simulation security in the presence of a malicious adversary. To capture the setting when the CRS authority is malicious, we require the following two security properties:

- **Accountability.** Suppose the authority generated a CRS maliciously. At a later point in time, it offers a service to recover the honest parties' inputs from the transcripts of protocol executions between these parties, using the trapdoors it embedded in the CRS. The accountability property guarantees that we can hold such a CRS authority accountable by producing a piece of publicly verifiable evidence that incriminates this authority for its malpractice. This evidence can then be presented in a court of law to penalize this authority. We formalize this by defining an efficient extractor that can interact[5] with this malicious authority and outputs a piece of evidence (a string). We associate with the scheme an algorithm Judge, which then determines whether this evidence is valid or not.
  The authority should not distinguish whether it interacts with the extractor (who is trying to incriminate the authority) or with a real party (who is trying to learn the inputs of the honest parties). Note that if the authority has some auxiliary information about the honest party's input, it can possibly produce the input without using the CRS trapdoor at all. In that case, it seems impossible to obtain incriminating evidence from the response of

[5] We stress that this extractor interacts with the malicious authority online *without being able to rewind the authority*. This is because, if we want to implicate the authority in the real world then we would not have the ability to rewind such an authority.

the authority. To avoid this issue, we specify a distribution $\mathcal{D}$ such that the inputs of the honest parties are sampled from this distribution in the security experiment. We stress that this requirement is only for the accountability security experiment. Our construction satisfies the usual definition of 2PC (without requiring any distribution on the inputs) in case the CRS is honest.

– **Defamation-free.** Of course, accountability cannot stand by itself. This notion opens up the possibility of falsely accusing even an honest CRS authority (who never partakes in running the input recovery service mentioned above) of malpractice. We complement the definition of accountability by defining another property called *defamation-free*. Roughly speaking, this definition states that just given an honestly generated CRS, it should be computationally infeasible to come up with an evidence that would incriminate an honest authority.

We study two variants of accountability. First, we study the scenario mentioned above in which two parties engage in a secure protocol. Then, one of them comes to the authority *after the fact* and asks to open the honest party's input. That party has to provide to the authority its view, which includes its own input and randomness. In the second (stronger) definition, we imagine the authority will be more cautious and refuse to answer such queries. Instead, the authority will insist on being involved from the beginning. In this model, the authority completely controls one of the parties, actively participates in the protocol execution on behalf of this party, and finally recovers and provides to this party the honest party's input. We refer to these notions as weak and strong accountability.

Impossibility Result. The first question is whether this new notion can be realized at all. Unfortunately, we show that even the weak definition cannot be realized for all functions. We show the following.

Theorem 1.1 (Informal)**.** *There exists a two-party functionality $\mathcal{F}$ such that there does not exist any secure two-party computation protocol for $\mathcal{F}$ in the CRS model satisfying both (weak) accountability and defamation-free properties.*

Specifically, the class of functionalities for which the above impossibility result hold are functionalities where given the output, we can efficiently recover the inputs. Indeed, an impossibility result is easy to see in this case since the authority can recover the input without even using any trapdoor related to the CRS (and in fact, anyone can recover the input of any party). Since this class of functionalities is somewhat trivial, and such functions are usually considered as functions where secure computation is not necessary (a trivial protocol where a party just gives its input suffices), this gives us hope that we can come up with positive results for large class of interesting functionalities. We focus on the setting of maliciously secure *two-round* two-party since that is known to be impossible to achieve in the plain model.

Construction. We then study the following class of (asymmetric) two-party functionalities $\mathcal{F}$: the two-party functionality takes as input (x, y) and outputs

$g(\{x_i\}_{y_i=1})$ to the second party (with input y) for some function g. That is, it outputs g on only those bits of x that are indexed by the bits of y set to 1. This class of functions includes for instance, oblivious transfer, private information retrieval, subset sum, and more. We show the following:

Theorem 1.2 (Informal)**.** *Assuming SXDH (Symmetric External Diffie-Hellman) on bilinear maps, there exists a two-round maliciously secure two-party computation protocol for $\mathcal{F}$ satisfying both weak accountability (with respect to the uniform distribution over the inputs) and defamation-free.*

Indeed, obtaining such a construction turns out to be surprisingly non-trivial and requires one to overcome novel technical challenges. We refer the reader to Section 2 (Technical Overview) for a summary of techniques.

Strong accountability for Oblivious Transfer. As mentioned, we study weak and strong accountability, depending on whether the malicious authority actively participates in the protocol execution or not. We focus on the oblivious transfer functionality and demonstrate that strong accountability is possible to achieve, based on a (seemingly) stronger assumption.

Theorem 1.3 (Informal)**.** *Assuming indistinguishability obfuscation for P/poly [5, 19] and SXDH in bilinear groups, there exists a two-round maliciously secure oblivious transfer protocol in the CRS model satisfying both strong accountability and defamation-free properties, with respect to the uniform distribution over the inputs.*

The techniques developed in the above construction can potentially be extended also for the class of functions in $\mathcal{F}$ for which Theorem 1.2 holds, although we focused on oblivious transfer for simplicity.

Non-Interactive Zero-Knowledge (NIZK). Another basic cryptographic primitive which relies on a CRS is NIZK. Indeed, CRS shows up in several cryptographic constructions primarily because they use NIZK as a building block. Similar to the 2PC case, we require the same guarantees as a regular NIZK when the CRS is honestly generated, namely, completeness, soundness and zero-knowledge. We associate with the proof system a Judge algorithm and require accountability and defamation free properties:

- **Accountability.** For any CRS^* that might be maliciously produced by the CRS authority, if there exists an adversary that upon receiving pairs (x, π) can recover the witness w, (where x is an instance, π is a proof that x is in the associated language, and w is the secret witness), then there exists an extractor that can create a piece of evidence τ that is accepted by Judge.
  As before, our accountability property is parameterized by a distribution $\mathcal{D}$ defined on the instance-witness pairs. Indeed this is necessary since if the authority can guess the witness without using the CRS trapdoor, the security guarantees we have in mind are impossible to achieve.
- **Defamation-free.** This states that no non-uniform probabilistic polynomial-time adversary $\mathcal{A}$ upon receiving a CRS that was honestly generated can come up with a piece of evidence τ that makes Judge accept.

We consider a NP language L consisting of instances of the form $(C, c_1, \ldots, c_m, b)$ such that $C : \{0,1\}^m \to \{0,1\}$ is a boolean circuit, each c_i is a commitment of x_i and moreover, $C(x_1, \ldots, x_n) = b$. We note that we can reduce any NP-complete language to this language based on the existence of rerandomizable commitments. We show the following.

Theorem 1.4 (Informal)**.** *Assuming SXDH on bilinear maps, there exists a NIZK for L in the CRS model satisfying both the accountability and the defamation-free properties.*

We can handle a class of distributions $\mathcal{D}$ with the only requirement being that a distribution in this class computes the commitments using uniform randomness while the circuit C and inputs $(x_1, \ldots, x_n)$ can be arbitrarily chosen.

Open Problems. We believe that a systematic study of the notions of accountability in the CRS model is an exciting line of research. Our work leaves open several natural questions. Can we broaden and characterize the class of functionalities for which accountable 2PC can be achieved? Can we extend our construction to more than two parties? While we focus on two rounds, obtaining even three round constructions would be valuable since the best known constructions in the plain model require at least four rounds.

One could also consider stronger notions where the authority only supplies some information about the input (e.g. the first bit of the input) rather than the entire input. In this setting, it seems the extractor would need to obtain multiple responses from the authority and somehow combine them into a single proof. Furthermore, while we focus on privacy (of the input in case of secure computation, or the witness in case of NIZK) in this work, what if the authority instead attacks correctness or soundness?

Another interesting direction is to consider other settings where CRS is used such as obtaining UC security.

Related Works. Our notion is inspired, in part, by broadcast encryption with traitor tracing [14] where, given a decryption box, there is a trace algorithm (similar to our Judge algorithm) which identifies the cheating party. However there are crucial differences. Our Judge algorithm does not have direct access to the CRS authority and only gets to see a string produced by the extractor. Furthermore, our extractor only gets to interact with the CRS authority *online* and, in particular, *does not get to rewind the CRS authority.* In another related line of research, Goyal [23] introduced what is known as accountable authority (AA) identity-based encryption (IBE) where if the authority generating the IBE public parameters is dishonest and releases a "decryption box", the authority can be implicated in a court. Our definition is also inspired by public verifiability in covert security, introduced by Asharov and Orlandi [3]. This definition shows how one can extract, given a transcript of the protocol, a piece of evidence showing that there was a misbehavior in the execution. The definition also requires defamation free, so that innocents cannot be implicated.

Another related notion of our work is subversion security, suggested by Bellare, Fuchsbauer, and Scafuro [7] (see also [18]). The work studies the security

of NIZKs in the presence of a maliciously chosen common reference string. It shows that several security properties can still be preserved when the CRS is maliciously generated. Nevertheless, it is shown that zero-knowledge cannot be preserved simultaneously with soundness when the common reference string is maliciously generated. Our work takes a different approach and seeks accountability when such misbehavior is detected. It will be intriguing to see how these two notions can intertwine.

2 Technical Overview

We start by explaining the main idea that underpins the constructions of NIZK, oblivious transfer and secure two-party computation with malicious authority security. Realizing this insight will lead us to different challenges in the context of designing each of the different primitives; we discuss the challenges for each of these primitives separately.

Main Idea. Our main idea is to force the CRS authority to include a transcript of execution of the protocol as part of the CRS, where the transcript has a secret x embedded inside. In the context of NIZKs, we force the authority to include a NIZK proof in the CRS where the witness contains the secret x. In the case of oblivious transfer and secure two-party computation, the sender and the receiver's input in the transcript are generated as a function of x.

In the honest execution, the transcript in the CRS is ignored. However, to argue accountability, the extractor will cleverly maul this transcript in the CRS to generate another transcript in such a way that the mauled transcript now has the embedded secret $x \oplus y$, where y is sampled by the extractor. The extractor then sends this mauled transcript to the malicious CRS authority. Since this authority offers a service to recover the inputs of the honest parties (or witness in the context of NIZKs), it recovers $x \oplus y$ and outputs this. Note that the authority was tricked into recovering an input that was in reality related to the secret that it hardwired inside the CRS. Now, the extractor has $x \oplus y$, and it can easily recover x. It recovers x and presents it as evidence to implicate the authority. The extractor could never recover x by itself without the "help" of the malicious authority, which also implies defamation free.

While this initial idea sounds promising, its realization involves technical challenges. We highlight some of them below.

- The first and foremost challenge is malleability. We hinged on the fact that the extractor can maul the transcript in the CRS. It turns out that malleability is a challenging problem. Malleability of transcripts has not been studied in the context of interactive protocols before and moreover, even in the setting of NIZKs, this has only been studied in the context of restricted relations.
- Another challenge is to ensure that the malicious authority cannot distinguish whether it is interacting with an extractor, who is trying to incriminate it, or is it interacting with a malicious party who only intends to learn

the inputs of the honest parties. In other words, we need an extractor who can produce transcripts that are computationally indistinguishable from the transcripts produced by real protocol executions (not the ones obtained by mauling the CRS).
- We mentioned above that we force the authority to include a transcript in the CRS. How do we ensure that the authority did indeed include a valid transcript in the CRS? The standard solution to employ a NIZK proof cannot work because the authority can violate soundness since it is the one who generates the CRS.
- Finally, to prove the defamation-free property, we need to argue that any probabilistic polynomial time adversary, (no matter how hard it tries) cannot come up with an evidence to implicate an honest authority. In other words, given the transcript in the CRS, it should be computationally infeasible to recover the secret x.

We now show how to implement our main idea, to construct NIZKs, oblivious transfer and secure 2PC with malicious authority security, and in the process we also discuss how to address the above challenges.

2.1 Malicious Authority Security for NIZK

We start by describing the NP relation associated with the proof system.

NP relation. Every instance in this relation is of the form $(C, c_1, \ldots, c_m, b)$, consisting of three components: (1) A boolean circuit $C : \{0,1\}^m \to \{0,1\}$; (2) Committed input $\mathbf{c} = (c_1, \ldots, c_m)$ hiding some bits $(x_1, \ldots, x_m)$ using decommitments $(r_1, \ldots, r_m)$; (3) A bit b satisfying $b = C(x_1, \ldots, x_m)$. The witness is therefore the bits $(x_1, \ldots, x_m)$ and their associated decommitments $(r_1, \ldots, r_m)$. In particular, embedding the commitments in the language guarantees average case hardness, as opposed to regular circuit satisfiability that might have only worst case hardness.

Base proof system. We start with a NIZK proof system and then modify this system to satisfy the desired properties. The proof system is obtained by employing the standard FLS trick [17].

To prove an instance $(C, c_1, \ldots, c_m, b)$ using a witness $\mathbf{x} = (x_1, \ldots, x_m)$ and de-commitments $\mathbf{r} = (r_1, \ldots, r_m)$, we simply use a NIWI proof system in which the prover can show that either it knows the witness $(\mathbf{x}, \mathbf{r})$, or that it knows a seed $\mathsf{s_{in}}$ for some string y that appears in the CRS, i.e., $y = \mathsf{PRG}(\mathsf{s_{in}})$. When the CRS is honestly generated, with overwhelming probability, such a pre-image does not exist, and thus the proof system is sound. Moreover, the simulator can generate an indistinguishable CRS in which $y = \mathsf{PRG}(\mathsf{s_{in}})$ for some trapdoor $\mathsf{s_{in}}$, enabling it to provide proofs without knowing the witnesses.

Accountability. To achieve accountability, we need to provide more information in the CRS. As a warmup, we will include the commitments $\mathbf{c}^0 = \mathsf{Com}(0; \mathbf{r}^0)$ and $\mathbf{c}^1 = \mathsf{Com}(1; \mathbf{r}^1)$ for random $\mathbf{r}^0, \mathbf{r}^1$. To prove accountability, we define an extractor who first samples a circuit C and a string $\mathbf{x} = (x_1 \cdots x_n)$ according

to the distribution of the honest prover, chooses a subset of the commitments $(\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_n})$, where $\mathbf{c}^{x_i}$ is a commitment of the bit x_i and is taken from the extra information in the CRS. Then, the extractor computes a proof π on the instance $(C, (\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_n}), C(\mathbf{x}))$. The authority, given the instance and the proof, will output the witness $(\mathbf{x}, \mathbf{r}^{x_1}, \ldots, \mathbf{r}^{x_n})$. This witness itself serves as an evidence that can be used to incriminate the authority; this is because it can be publicly verified that $(x_i, \mathbf{r}^{x_i})$ is a valid opening for $\mathbf{c}^{x_i}$. Moreover, just given the CRS, it is computationally infeasible to produce an opening; thus, defamation-free is guaranteed as well.

There are four major issues with this approach. The *first* issue is the following: the authority upon recovering the witness $(\mathbf{x}, \mathbf{r}^{x_1}, \ldots, \mathbf{r}^{x_n})$, or even by just viewing the instance $(C, (\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_n}), C(\mathbf{x}))$ to be opened, will realize that it corresponds to the randomness associated with the commitments in the CRS. So, it will be able to figure out that it is the extractor who submitted the proof. The *second* issue is that it is unclear how the extractor will be able to produce a valid proof on the instance $(\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_n})$. Indeed, the binding property of the commitment scheme and the soundness of the NIWI proof tell us that this should not be possible. The *third* issue is that the authority is the one who generates the public parameters of the commitment scheme, and might generate them as computationally binding instead of perfectly binding. This means that even if the authority opens the input, it might open $\mathbf{c}^{x_i}$ to $1 - x_i$. In this case, it is unclear how to implicate the authority. Finally, the *fourth* issue is that we need to verify that the malicious authority included commitments of 0 and 1 only.

- To get around the first issue, we use a rerandomizable commitment scheme. Given commitment to a message m, we can rerandomize this commitment in such a way that randomness of the new commitment information-theoretically hides the randomness used in the old commitment. To see why this is useful, note that the extractor can rerandomize the commitments $(\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_n})$. Now, the randomness recovered by the authority is identically distributed to fresh commitments of $(x_1, ..., x_n)$.
- To get around the second issue, we add to the language of our NIWI a third branch: given a statement $(C, \mathbf{c}_1, \ldots, \mathbf{c}_m, b)$, the commitments $(\mathbf{c}_1, \ldots, \mathbf{c}_m)$ were obtained as re-randomizations of the two commitments $\mathbf{c}^0$ and $\mathbf{c}^1$ in the CRS, and the extractor has to provide the re-randomization information. Thus, to generate an implicating transcript, the extractor chooses any circuit C and input $(x_1, \ldots, x_m)$ according to the distribution of the honest prover. Moreover, it evaluates $C(x_1, \ldots, x_m) = b$, re-randomizes the commitments $\mathbf{c}^{x_1}, \ldots, \mathbf{c}^{x_m}$ to obtain $(\mathbf{c}_1, \ldots, \mathbf{c}_m)$. The extractor then gets an instance $(C, \mathbf{c}_1, \ldots, \mathbf{c}_m, b)$ with a proof π_{NIWI}.
- For the third issue, we show that it does not matter whether the commitment $\mathbf{c}^{x_i}$ is opened to x_i or $1 - x_i$. In particular we show that, using the rerandomizability property of the commitment scheme, having either of the two openings is sufficient to implicate the authority.
- Finally, to overcome the forth issue, the authority will provide four commitments as part of the CRS, $\big((c_0^0, c_1^0), (c_0^1, c_1^1)\big)$, and prove using a NIWI proof

(which does not require CRS) that one of the branches (c_0^0, c_1^0) or (c_0^1, c_1^1) consists of commitments to 0 and 1. Before participating in proving any statement with respect to this CRS, one has to check using the proof provided in the CRS that the CRS is correctly computed, i.e., that there is a method to implicate the authority if corrupted.

To argue accountability with the modified CRS, the extractor needs to pick the correct branch to rerandomize. However, it does not know which is the right branch. Instead it samples one of the branches uniformly at random and proceeds. If we had the guarantee that the authority recovered the witness with non-negligible probability then we have the guarantee that the extractor still succeeds in coming up with an incriminating evidence with non-negligible probability.

We can argue defamation-free using the witness-indistinguishability property of the proof in the CRS in conjunction with the hiding property of the commitment scheme.

2.2 Malicious Authority Security for Oblivious Transfer

We now focus our attention on secure two party computation protocols. To gain better intuition and to understand the difficulties we cope with, we start with studying a specific functionality — oblivious transfer. The solutions and techniques developed in addressing this functionality will also be useful in understanding the general case.

As opposed to NIZK which consists of one message and only the prover has some private input (the witness), in oblivious transfer both parties have private inputs. We consider a parallel repetition of 1-out-of-2 bit oblivious transfer, in which the receiver holds a string $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n) \in \{0,1\}^n$, and the sender holds two messages $\mathbf{m}_0 = (m_1^0, \ldots, m_n^0)$, $\mathbf{m}_1 = (m_1^1, \ldots, m_n^1) \in \{0,1\}^n$. Only the receiver receives the output which is $m_1^{\sigma_1}, \ldots, m_n^{\sigma_n}$. A two-round protocol of oblivious transfer in the CRS model consists CRS generation algorithm GenCRS (run by the authority) that outputs $\mathsf{CRS}_{\mathsf{OT}}$, and two algorithms $\mathsf{OT}_1, \mathsf{OT}_2$ for generating the transcript. The receiver runs $\mathsf{msg}_R = \mathsf{OT}_1(\mathsf{CRS}_{\mathsf{OT}}, \boldsymbol{\sigma})$ to obtain the message msg_R from the receiver to the sender, followed by a message $\mathsf{msg}_S = \mathsf{OT}_2(\mathsf{CRS}_{\mathsf{OT}}, \mathbf{m}_0, \mathbf{m}_1, \mathsf{msg}_R)$ from the sender to the receiver. The receiver then makes some local computation to output $m_1^{\sigma_1}, \ldots, m_n^{\sigma_n}$.

As the functionality hides information for both the receiver $(m_1^{1-\sigma_1}, \ldots, m_n^{1-\sigma_n})$ and the sender $(\boldsymbol{\sigma})$, both parties might come to the malicious authority and ask to open the same transcript, while extracting different information from it. We, therefore, have to discuss two different scenarios and show that in either case, if the authority offers help to either of the two parties, it can be implicated. In the first scenario, which we call *malicious sender*, the sender submits its view to the authority and tries to learn the input of the receiver. We define *malicious receiver* analogously. We follow the same oblivious transfer protocol that is secure against malicious adversaries. As mentioned earlier in our discussion, to achieve the protocol's security, we cannot hope to prevent the malicious

authority from making any trapdoors in $\mathsf{CRS}_{\mathsf{OT}}$. Those trapdoors are essential ingredients when proving the simulation security of the protocol. All we can do is to prevent it from using that trapdoor, and specifically from divulging secrets to others.

Dealing with a malicious sender. Following our general template, the authority generates $\mathsf{CRS}_{\mathsf{OT}}$. Moreover, it randomly samples a challenge $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)$, for some sufficiently long m, and appends $f(\boldsymbol{\sigma})$ to the CRS, where f is a one-way function. Then, we want to embed $\boldsymbol{\sigma}$ in transcript of OT, i.e., add $\mathsf{msg}_R^{\boldsymbol{\sigma}} = \mathsf{OT}_1(\boldsymbol{\sigma})$. Without knowing the receiver's randomness, no one can learn $\boldsymbol{\sigma}$ just from seeing the message $\mathsf{OT}_1(\boldsymbol{\sigma})$, as guaranteed from the receiver's security in the protocol against the malicious sender. This guarantees defamation free. Yet, our goal is give the extractor the ability to maul those transcripts such that if ever opened, we will have a piece of evidence to implicate the authority.

A natural idea is to just complete the transcript with any $\mathbf{m}_0, \mathbf{m}_1$ to obtain $\mathsf{msg}_S = \mathsf{OT}_2(\mathsf{CRS}_{\mathsf{OT}}, \mathbf{m}_0, \mathbf{m}_1, \mathsf{msg}_R^{\boldsymbol{\sigma}})$. Then, to come up with the pair $(\mathsf{msg}_S, \mathsf{msg}_R)$ to the authority. But, the authority will refuse to open such a transcript – it can clearly identify that the receiver's secret input is $\boldsymbol{\sigma}$, i.e., the secret challenge it generated! Moreover, it can identify that the message msg_R in the transcript is identical to the message it published in the CRS. We need a stronger method that enables us to complete transcripts to any input of the sender and to maul the receiver's input and re-randomizes it.

To achieve that, we again successfully avoid the issue of malleability using rerandomization and by adding more information in the CRS. Recall that the protocol is a parallel repetition of bit OT, i.e., $\mathsf{OT}_1(\boldsymbol{\sigma}) = \mathsf{OT}_1^{bit}(\sigma_1), \ldots, \mathsf{OT}_1^{bit}(\sigma_m)$, where $(\mathsf{OT}_1^{bit}, \mathsf{OT}_2^{bit})$ is the underlying bit OT protocol. The authority will have to generate for every bit two transcripts, $\alpha_{i,0} = \mathsf{OT}_1^{bit}(\sigma_i \oplus 0)$ and $\alpha_{i,1} = \mathsf{OT}_1^{bit}(\sigma_i \oplus 1)$. This enables the extractor to obtain a transcript for $\boldsymbol{\sigma} \oplus \Delta$ for the receiver for every $\Delta = (\Delta_1, \ldots, \Delta_n)$ of its choice, and any input $(\mathbf{m}_0, \mathbf{m}_1)$ of the sender of its choice. That is, to generate $\mathsf{OT}_1(\boldsymbol{\sigma} \oplus \Delta)$, do the following:

$$\mathsf{OT}_1(\boldsymbol{\sigma} \oplus \Delta) = (\alpha_{1,\Delta_1}, \ldots, \alpha_{n,\Delta_n}) = \left(\mathsf{OT}_1^{bit}(\sigma_1 \oplus \Delta_1), \ldots, \mathsf{OT}_1^{bit}(\sigma_n \oplus \Delta_n)\right) .$$

It re-randomizes each one of these messages, and completes it to full transcript with any messages $\mathbf{m}_0, \mathbf{m}_1$ of its choice. The authority receiving such a transcript has no way to tell that this transcript was generated using the transcripts it published in the CRS. By extracting the input of the receiver it discloses itself.

Dealing with a malicious receiver. Following a similar approach, recall that on input $(\mathbf{m}_0, \mathbf{m}_1)$ for the sender and $\boldsymbol{\sigma}$ for the receiver, the oblivious transfer functionality hides only $(m_1^{1-\sigma_1}, \ldots, m_n^{1-\sigma_n})$ but reveals $(m_1^{\sigma_1}, \ldots, m_n^{\sigma_n})$ to the receiver. Therefore, it seems natural to embed the challenge in the hidden part of the message. The authority chooses a new challenge $\mathbf{x} = (x_1, \ldots, x_n)$ and publishes $f(\mathbf{x})$ in the CRS. Moreover, it creates transcripts that correspond to $\mathbf{x}$ and enables the extractor to produce transcript for every input $\mathbf{r} = (r_1, \ldots, r_n)$

of the receiver and "shift" $\Delta = (\Delta_1, \ldots, \Delta_n)$, while embedding x_i in position $1 - r_i$ (which is not revealed), to obtain $((x_1 \oplus \Delta_1)^{1-r_1}, \ldots, (x_n \oplus \Delta_n)^{1-r_n})$. If the malicious authority ever opens the transcript, that is, it recovers $((x_1 \oplus \Delta_1)^{1-r_1}, \ldots, (x_n \oplus \Delta_n)^{1-r_n})$, it can then extract $\mathbf{x}$ from this (since it knows the "shift") and implicate the authority. To do that, first observe that there are 8 possible transcripts for each bit OT: the input of the receiver is $r_i \in \{0,1\}$ and the input of the sender is $(m_0, m_1) \in \{0,1\}^2$. To enable the extractor to generate any transcript it wishes, for every bit-OT $i \in \{1, \ldots, n\}$, the CRS authority is expected to produce (as part of the CRS generation) four values as follows: For every $\mu, \Delta_i \in \{0,1\}$:

$$\beta^i_{0,\mu,\Delta_i} = \mathsf{OT}(0, (\mu, x_i \oplus \Delta_i)) \quad \text{and} \quad \beta^i_{1,\mu,\Delta_i} = \mathsf{OT}(1, (x_i \oplus \Delta_i, \mu)) \ ,$$

while $\mathsf{OT}(\sigma, (m_0, m_1))$ denotes a full transcript of a bit OT where the input of the receiver is σ and the sender is (m_0, m_1), and we omit $\mathsf{CRS}_{\mathsf{OT}}$ for brevity. Observe that for each $i \in \{1, \ldots, n\}$ of the bit-OTs provided in the CRS, the bit x_i is not revealed in the transcript, as it corresponds to the input of the sender that is not revealed. On the other hand, the randomness and the input of the receiver is given in the clear.

The extractor can now choose any message $\mathbf{m} = (m_1, \ldots, m_n)$ of its choice and any $\Delta = (\Delta_1, \ldots, \Delta_n)$, and for every input $\mathbf{r} = (r_1, \ldots, r_n)$ of the receiver, it can generate a transcript $(\beta^1_{r_1,m_1,\Delta_1}, \ldots, \beta^n_{r_n,m_n,\Delta_n})$, which embeds a masking of $\mathbf{x}$. The extractor rerandomizes this transcript and, if opened by the authority, the extractor can easily recover $\mathbf{x}$.

Finally, as the authority has to produce many transcripts that are correlated with the challenge, it has to prove that it generated all of them as specified. Just as in malicious authority security for NIZK, we double all the new information in the CRS and ask it to prove using a NIWI that one of the branches was generated as specified.

Re-randomizable oblivious transfer. As mentioned above, to allow this to work, we must ensure that the oblivious transfer transcript is *rerandomizable*. Informally, we say that an OT transcript is rerandomizable if given a transcript of execution of OT, we should be able to transform into another transcript on the same inputs. The rerandomization guarantee is that even given the secret randomness and the input of both parties in the original transcript, a distinguisher receiving a view of one of the parties should not be able to figure out whether the view comes from a new transcript (with the same inputs), or the view was rerandomized. We show that the oblivious transfer protocol of Peikert, Vaikuntanathan, and Waters [31] is a perfect fit for our needs: it is a two-round oblivious transfer in the CRS model, and we augment the protocol with rerandomization procedures.

Strong accountability. The protocol described above works when the malicious authority does not participate actively in the protocol execution. To understand why, we first remark that for a malicious receiver, the authority provides transcripts where the input and randomness of the receiver are in the clear (i.e., provides the complete view of the receiver).

However, in strong accountability, the extractor now talks directly to the adversary. It receives the first message in the protocol from the adversary, and it cannot know the randomness and the private input of the adversary. Matching the correct transcript from the CRS to the one just received by the adversary is impossible, due to the receiver's privacy. Specifically, to embed x_i in position $1 - r_i$, we have to know r_i.

On the other hand, this problem does not occur for the malicious sender's case, as it sends the second message in the protocol. In fact, the above-described method already achieves strong accountability against malicious sender.

Achieving strong accountability via indistinguishability obfuscation. We first start with the following idea. As now the transcripts are given "on the fly" to the extractor, we do not even try to embed the secret challenge into the transcript. Instead, we generate an honestly generated transcript using random messages $\mathbf{m}_0, \mathbf{m}_1$ that the extractor itself *does not even know*, and give it also $\mathbf{m}_0 \oplus \mathbf{m}_1 \oplus \mathbf{x}$. Now the bits $(m_1^{1-r_1}, \ldots, m_n^{1-r_n})$ are not known to the sender and the receiver, and thus $\mathbf{x}$ is protected using one-time pad.

To implement this idea, we use indistinguishability obfuscation[6]. The authority obfuscates a circuit C that on input msg_R generates msg_S on random inputs $\mathbf{m}_0, \mathbf{m}_1$ and gives also $\mathbf{m}_0 \oplus \mathbf{m}_1 \oplus \mathbf{x}$. Crucially, the evaluator of the circuit (the extractor) does not know $\mathbf{m}_0, \mathbf{m}_1$. The messages $\mathbf{m}_0, \mathbf{m}_1$ are generated using a pseudorandom key chosen by the CRS authority and was hardwired in the circuit. Moreover, the defamation free proof is now rather involved as it requires showing that the hardwired value $\mathbf{x}$ is not revealed even though it is also hardwired in the circuit.

This description is too simplified and is not sound. If the extraction just sends the message msg_R as received from the obfuscated circuit, the authority can clearly identify it as it had generated the obfuscated circuit. The extractor therefore has to rerandomize the message it receives as output from the circuit. But this still does not suffice, as the authority can also identify that two messages $\mathbf{m}_0, \mathbf{m}_1$ were generated by the circuit, as those are pseudorandom and it knows the key used to generate them. Therefore, we modify the circuit such that given a message msg_R it generates four different transcripts, i.e.,

$$\beta_{\tau_0,\tau_1} = \mathsf{OT}_2(\mathbf{m}_0 \oplus \tau_0^n, \mathbf{m}_1 \oplus \tau_1^n, \mathsf{msg}_R)$$

for every $\tau_0, \tau_1 \in \{0, 1\}$. This enables the extractor to pick any masking it wishes to $\mathbf{m}_0, \mathbf{m}_1$, similarly to the case of weak accountability. Whenever the authority opens such a message, the extractor recovers $\mathbf{m}_0, \mathbf{m}_1$ and thus also $\mathbf{x}$.

[6] An indistinguishability obfuscator [5, 19] is a compiler that on input circuit C outputs a functionally equivalent circuit $\widehat{C}$. Moreover, it gurantees that the obfuscations of two functionally equivalent circuits (of the same size) are computationally indistinguishable.

2.3 Malicious Authority Security for Two Party Computation

We now focus our attention on general purpose secure two party computation protocols. We first start with an overview that shows that it is not possible to achieve general purpose secure computation protocols for all functions. We then complement our negative result by identifying a class of functionalities (which subsumes oblivious transfer functionalities) for which we can achieve a positive result.

Impossibility result. When it comes to tackling the problem of designing secure computation for general functionalities, we realize that we cannot simultaneously achieve accountability and defamation-free properties. Since the authority recovers the inputs of the honest parties, it has to be the case that the output of the functionality does not trivially leak the inputs, i.e., each party really needs the help of the authority to recover the inputs of the honest parties. When considering functions that do not provide such secrecy, we show that there does not exists a protocol that can address malicious authority security. Luckily, those functions, at least intuitively, are the functions for which secure computation is not necessary to begin with. We formalize this intuition and show that it is impossible to achieve secure two party computation for any functionality, as there exists functions for which achieving malicious authority security is impossible.

Observe also that so far, both in the NIZK example and in oblivious transfer, the functionality hides sufficient information (in NIZK this is the witness w; in oblivious transfer these are the choice bits of the receiver and for the sender, those are the bits in the input that were not selected by the receiver). This is very intuitive: If the function is not hiding, then there is no need for help from the authority to recover the private inputs.

Positive result. We therefore restrict our attention to a specific class of functions $\mathcal{F}$ that is guaranteed to have some form of secrecy. We also focus on the asymmetric case where only one of the parties (designated as the receiver) gets an output. Specifically we look at functions in which the inputs of both parties is sufficiently long, and for every input of the receiver, there exists at least λ bits in the inputs of the sender that are not meaningful and do not affect the output, where λ denotes the security parameter. Interesting functions that are captured in this class are oblivious transfer, private information retrieval, subset sum, and more.

We describe a two-round secure computation protocol for computing all the functions in the family in the CRS model, and then show how to enhance its security to achieve malicious authority security.

The base protocol is a standard two-round two-party secure computation protocol that combines two-round oblivious transfer with garbled circuits. Denoting the sender's input as $\mathbf{x} = (x_1, \ldots, x_\ell)$ and the receiver's input as $\mathbf{y} = (y_1, \ldots, y_\ell)$, the receiver's first message is simply $\mathsf{msg}_R = \mathsf{OT}_1(\mathbf{y})$. The second message of the sender is a bit more involved, and this complication comes to accommodate malicious authority security at a later stage. Given a circuit C for computing the function $F \in \mathcal{F}$, the sender generates a garbled circuit GC together with

the labels $K^{(x)}_{i,0}, K^{(x)}_{i,1}$ for each input of the sender, and labels $K^{(y)}_{i,0}, K^{(y)}_{i,1}$ for each input of the receiver. To receive the output on $\mathbf{x}, \mathbf{y}$, the receiver has to obtain the labels $K^{(x)}_{i,x_i}, K^{(y)}_{i,y_i}$ and output $\mathsf{Eval}(\mathsf{GC}, K^{(x)}_{i,x_i}, K^{(y)}_{i,y_i})$. To do that, the "classic" approach is to instruct the sender to send the following message:

$$\{K^{(x)}_{i,x_i}\}_{i\in[\ell]}, \quad \mathsf{OT}_2\left((K^{(y)}_{i,0})_{i\in[\ell]}, (K^{(y)}_{i,1})_{i\in\ell}, \mathsf{msg}_R\right) , \tag{1}$$

i.e., sending the labels that correspond to the sender's input, and the labels correspond to the receiver's input are obtained using the oblivious transfer. For the generation of the transcript by the extractor, it would be easier to send the labels of the sender also in an OT message. We instead send the following message:

$$\mathsf{OT}_2\left((K^{(x)}_{i,0} \parallel K^{(y)}_{i,0})_{i\in[\ell]}, \quad (K^{(x)}_{i,x_i} \| K^{(y)}_{i,1})_{i\in\ell}, \ \mathsf{msg}_R\right) . \tag{2}$$

That is, the receiver always receives labels that corresponds to its input, as before. As for the labels that correspond to the input of the sender, in case $y_i = 1$, then the receiver obtains $K^{(x)}_{i,x_i}$, i.e., the "correct" label. On the other hand, in case $y_i = 0$ then the receiver obtains $K^{(x)}_{i,0}$, regardless of what the input of the sender is, i.e., it receives a label that corresponds to $x_i = 0$. This still guarantees correctness as in the case where $y_i = 0$, the input x_i does not affect the output, and the evaluation of the circuit when $x_i = 0$ and $x_i = 1$ gives the same result. Here we rely on the structure of functions that we compute. Together with this OT_2 message, the sender also sends a NIWI proof that it either generated the message correctly as instructed, or it knows some trapdoor in the CRS.

Enhancing to malicious authority security. Since the first message in the protocol of the receiver is just $\mathsf{OT}_1(\mathbf{y})$, this case reduces to the case of obtaining malicious authority security in the case of a malicious receiver in oblivious transfer.

For the case of malicious sender, we again follow our general template and the CRS authority chooses a random challenge $\mathbf{r} = (r_1, \ldots, r_\lambda) \in \{0,1\}^\lambda$, and gives out $f(\mathbf{r})$. The goal now is to find what transcripts to provide such that the extractor will be able to embed $\mathbf{r}$ to the input of the sender. If we would have sent the keys as described in Eq. (1), then to let the extractor choose inputs of the sender as a function of $\mathbf{r}$, the authority would have to give both keys $K^{(x)}_{i,r_i}, K^{(x)}_{i,1-r_i}$. This implies that the extractor will be able to evaluate the function on both values of r_i, breaking defamation free. Embedding the changes inside the OT_2 message enables us to maul that message without giving away any information about $\mathbf{r}$. Specifically, we always maul the part that the receiver did not ask for, similarly to the way it is done for oblivious transfer.

Organization. The remaining of the paper is organized as follows. We provide the necessary preliminaries in Section 3. In Section 4 we formally define our new notion. In Section 5 we provide the construction of NIZK. Due to lack of space, the construction of oblivious transfer and the two-party computation are deferred to the full version of this paper.

3 Preliminaries

Notation and conventions. We let λ denote the security parameter. We let $[n]$ denote the set $\{1, \ldots, n\}$. We use PPT as shorthand for probabilistic polynomial time. A function μ is negligible if for every positive polynomial $p(\cdot)$ and all sufficiently large λ's, it holds that $\mu(\lambda) < 1/p(\lambda)$.

A probability ensemble $X = \{X(a, \lambda)\}_{a \in \{0,1\}^*; \lambda \in \mathbb{N}}$ is an infinite sequence of random variables indexed by $a \in \{0,1\}^*$ and λ. In the context of zero knowledge, the value a will represent the parties' inputs and λ will represent the security parameter. All parties are assumed to run in time that is polynomial in the security parameter. Two probability ensembles $X = \{X(a, \lambda)\}_{a \in \{0,1\}^*; \lambda \in \mathbb{N}}$, $Y = \{Y(a, \lambda)\}_{a \in \{0,1\}^*; \lambda \in \mathbb{N}}$ are are said to be computationally indistinguishable, denoted by $X \approx_c Y$, if for every non-uniform polynomial-time algorithm D there exists a negligible function μ such that for every $a \in \{0,1\}^*$ and every $\lambda \in \mathbb{N}$,

$$|\Pr[D(X(a, \lambda)) = 1] - \Pr[D(Y(a, \lambda)) = 1]| \leq \mu(\lambda)$$

We denote by $x \leftarrow D$ a sampling of an instance x according to the distribution D.

We denote vectors using a **bold** font, e.g., $\boldsymbol{\alpha} \in \{0,1\}^n$, when it is usually clear from context what is the vector size. We let $\boldsymbol{\alpha}[i]$ denote the ith coordinate of the vector.

3.1 Rerandomizable Commitment Scheme

Commitment scheme is a basic tool in cryptographic protocols. Informally, we require the commitment scheme to satisfy two properties, the first is called perfect binding, which means that the sets of all commitments to different values are disjoint; for all $x \neq x'$ it holds that $\mathsf{Com}(x) \cap \mathsf{Com}(x') = \emptyset$ where $\mathsf{Com}(x) = \{c \mid \exists r \text{ such that } \mathrm{c} = \mathsf{Com}(\mathrm{x}; \mathrm{r})\}$ and $\mathsf{Com}(x') = \{c \mid \exists r \text{ such that } \mathrm{c} = \mathsf{Com}(\mathrm{x}'; \mathrm{r})\}$. The second property is computational hiding; which means that the commitments to different strings are computationally indistinguishable.

In addition to the perfect binding and computational hiding properties, we also require the commitment scheme to be rerandomizable. The commitment scheme $\mathsf{C} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Rerand}, f_{\mathsf{com}})$ has the following syntax and properties:

- $\mathsf{p} \leftarrow \mathsf{Setup}(1^\lambda)$: outputs some public parameters p. Let the message space be $\mathcal{M}$ and the commitment space be $\mathcal{C}$.
- $\mathbf{c} \leftarrow \mathsf{Com}(\mathsf{p}, m; r)$: The algorithm gets $m \in \mathcal{M}$ and outputs a commitment $c \in \mathcal{C}$. The opening of the commitment is simply r.
- $\mathbf{c}' \leftarrow \mathsf{Rerand}(\mathsf{p}, \mathbf{c}; s)$: On input parameters p, commitment $\mathbf{c}$ and randomness s, Rerand outputs a randomized commitment $\mathbf{c}'$ to the same value. Moreover, we require the existence of an efficient function f_{com} such that for any randomness m, r, s the following holds:
  - $\mathsf{Rerand}(\mathsf{p}, \mathsf{Com}(\mathsf{p}, m; r); s) = \mathsf{Com}(\mathsf{p}, m, s')$ where $s' = f_{\mathsf{com}}(r, s)$.

Moreover, it is required that for every fixed s, the function $f_{\mathsf{com}}(\cdot, s)$ is bijection, and for every r, the function $f_{\mathsf{com}}(r, \cdot)$ is a bijection as well. In particular, this means that given s', s one can find r for which $s' = f_{\mathsf{com}}(r, s)$.

Such a scheme can be constructed from the DLIN assumption, as showed in [26]. It's rerandomization properties were discussed in [2].

3.2 Non-Interactive Zero Knowledge (NIZK)

Let L be an NP language and let R_L be its associated relation. For $(x, w) \in R_L$ we sometimes denote x the statement and w its associated witness.

Definition 3.1. *Let $L \in \mathsf{NP}$ and let R_L be the corresponding NP relation. A triple of algorithms $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ is called* non interactive zero knowledge (NIZK) argument *for L if it satisfies:*

- **Perfect completeness:** *For all security parameters $\lambda \in \mathbb{N}$ and for all $(x, w) \in R_L$,*

$$\Pr\left[\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda);\ \pi \leftarrow \mathsf{Prove}(\mathsf{CRS}, x, w) : \mathsf{Verify}(\mathsf{CRS}, x, \pi) = 1\right] = 1$$

- **Adaptive Soundness:** *For all prover P^*, there exists a negligible function μ such that for all λ:*

$$\Pr\left[\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda); (x, \pi) \leftarrow P^*(\mathsf{CRS}) : \mathsf{Verify}(\mathsf{CRS}, x, \pi) \wedge x \notin L\right] \leq \mu(\lambda)$$

  When this probability is 0, we say that Π is perfectly sound.

- **Adaptive Zero Knowledge:** *There exists a PPT simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ where $\mathcal{S}_1(1^\lambda)$ outputs $(\mathsf{CRS}_{\mathcal{S}}, \tau)$ and $\mathcal{S}_2(\mathsf{CRS}_S, \tau, x)$ outputs $\pi_{\mathcal{S}}$ such that for all non-uniform PPT adversaries $\mathcal{A}$,*

$$\left\{\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)\ :\ \mathcal{A}^{\mathcal{O}_1(\mathsf{CRS}, \cdot, \cdot)}(\mathsf{CRS})\right\}$$
$$\approx_c \left\{(\mathsf{CRS}_{\mathcal{S}}, \tau) \leftarrow \mathcal{S}_1(1^\lambda)\ :\ \mathcal{A}^{\mathcal{O}_2(\mathsf{CRS}, \tau, \cdot, \cdot)}(\mathsf{CRS}_{\mathcal{S}})\right\}$$

  where $\mathcal{O}_1, \mathcal{O}_2$ on input (x, w) first check that $(x, w) \in R_L$, else output $\perp$. Otherwise, $\mathcal{O}_1$ outputs $\mathsf{Prove}(\mathsf{CRS}, x, w)$ and $\mathcal{O}_2$ outputs $\mathcal{S}_2(\mathsf{CRS}_{\mathcal{S}}, \tau, x)$.

3.3 Non-Interactive Witness Indistinguishability (NIWI)

One building block that we often use in our construction is non-interactive witness indistinguishability. It is useful for our purposes as it does not require a common-reference string. NIWI in the plain model can be constructed based on the DLIN assumption [26] and can be constructed assuming either trapdoor permutations and derandomization assumptions [6].

Definition 3.2. *A pair of PPT algorithms $(\mathsf{Prove}, \mathsf{Verify})$ is a NIWI for an NP relation R_L if it satisfies:*

1. **Completeness:** *For every* $(x, w) \in R_L$,

$$\Pr\left[\mathsf{Verify}(x, w) = 1 \ : \ \pi \leftarrow \mathsf{Prove}(x, w)\right] = 1 \ .$$

2. **Soundness:** *There exists a negligible function* μ *such that for every* $x \notin L$ *and* $\pi \in \{0,1\}^*$*:*

$$\Pr\left[\mathsf{Verify}(x, \pi) = 1\right] \leq \mu(|x|) \ .$$

3. **Witness indistinguishability:** *For any sequence* $\{(x, w_1, w_2) : w_1, w_2 \in R_L(x)\} \in \mathcal{I}$*:*

$$\{\pi_1 : \pi_1 \leftarrow \mathsf{Prove}(x, w_1)\}_{(x,w_1,w_2)\in\mathcal{I}} \approx_c \{\pi_2 : \pi_2 \leftarrow \mathsf{Prove}(x, w_2)\}_{(x,w_1,w_2)\in\mathcal{I}} \ .$$

4 Defining Malicious Authority Security

In this section, we define the notion of malicious authority security for cryptographic protocols defined in the CRS model. There are two aspects to our definition, depending on whether CRS is honestly generated or if its generated by a malicious authority. In the first case, if the CRS is honestly generated, we require that the protocol satisfies the same traditional security requirements described in the literature. In the second case, suppose that the CRS is generated by a malicious authority and specifically, if the malicious authority runs a service that let the adversarial entities, participating in the protocol, to recover the inputs of the honest parties. In this setting, we should be able to implicate the malicious authority of its wrongdoing. Formally speaking, we define an extractor that interacts with the malicious authority and comes up with an evidence τ that can presented to a Judge, defined by a Judge algorithm, who verifies whether the presented evidence is valid. At the same time, we require the property that no efficiency adversary can present an evidence that can falsely accuse the honest authority of running a service.

We first discuss the definition of malicious authority security for NIZK (Section 4.1), and then extend the ideas to general secure two party computation (Section 4.2).

4.1 Malicious Authority Security for NIZK

We start by defining malicious authority security in the context of non-interactive zero-knowledge systems.

A NIZK system consists of a triplet of algorithms $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$. In addition, we define a PPT algorithm Judge, which will be necessary for malicious authority security.

- $b \leftarrow \mathsf{Judge}(\mathsf{CRS}, \tau)$ where $b \in \{\mathsf{honest}, \mathsf{corrupted}\}$: The algorithm receives as input the (possibly corrupted) CRS and some transcript τ, and outputs a b, indicating whether the τ proves that the CRS CRS is corrupted or not.

Definition 4.1. *Let* $L \in \mathsf{NP}$ *and let* R_L *be the corresponding* NP *relation. We say that a NIZK system* $\Pi = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Judge})$ *has malicious authority security with respect to distribution* $\mathcal{D}$ *if:*

1. *The system* $\Pi' = (\mathsf{GenCRS}, \mathsf{Prove}, \mathsf{Verify})$ *is a NIZK proof system for* L.
2. **(Accountability:)** *We say that the NIZK scheme* Π *achieves* accountability with respect to distribution $\mathcal{D}$ *if for all sufficiently large security parameter* $\lambda \in \mathbb{N}$*, any adversary* $\mathcal{A}$*, if there exists a non-negligible function* $\epsilon_1(\cdot)$ *such that*
$$\Pr[\mathsf{Acc.Real}_{\Pi,\mathcal{A},q}(\lambda) = 1] \geq \epsilon_1(\lambda)$$
then there exists a probabilistic polynomial time oracle-aided algorithm Ext *making at most* q *queries, and a non-negligible function* $\epsilon_2(\cdot)$ *such that:*
$$\Pr[\mathsf{Acc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda) = 1] \geq \epsilon_2(\lambda)$$
where the random variables $\mathsf{Acc.Real}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)$ *and* $\mathsf{Acc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)$ *are defined below.*

   $\underline{\mathsf{Acc.Real}_{\Pi,\mathcal{A},q}(\lambda)}$*:*
   The adversary $\mathcal{A}(1^\lambda)$ *outputs some* CRS^**, and we repeat the following for* q *iterations:*
   - $(x_i, w_i) \leftarrow \mathcal{D}$ *and then* $\pi_i \leftarrow \mathsf{Prove}(\mathsf{CRS}^*, (x_i, w_i))$.
   - *If* $\mathsf{Verify}(\mathsf{CRS}^*, x_i, \pi_i) \neq 1$ *then abort and the output of the experiment is* 0.
   - $\mathcal{A}$ *is given* (x_i, π_i).

   $\mathcal{A}$ *outputs some* (i, x_i', w_i') *for* $i \in [q]$*. The output of the experiment is* 1 *if* $x_i = x_i'$ *and* $R_L(x_i, w_i') = 1$.

   $\underline{\mathsf{Acc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)}$*:*
   The adversary $\mathcal{A}(1^\lambda)$ *outputs some* CRS^**, and we invoke the extractor* $\mathcal{E}$ *on input* (CRS^*)*. We run* q *iterations in which in each iteration* $\mathcal{E}$ *outputs some* (x_i, π_i) *that is forwarded to* $\mathcal{A}$*. After all iterations,* $\mathcal{A}$ *outputs some* (i, x_i', w_i') *for some* $i \in [q]$*. The extractor* $\mathcal{E}$ *then outputs* τ*, and the output of the experiment is* 1 *if* $\mathsf{Judge}(\mathsf{CRS}^*, \tau) = \mathsf{corrupted}$.
3. **(Defamation-free:)** *For every* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* $\mu(\cdot)$*, such that for all* λ*:*
$$\Pr\left[\mathsf{Judge}(\mathsf{CRS}, \mathcal{A}(\mathsf{CRS})) = \mathsf{corrupted}\right] \leq \mu(\lambda) \ ,$$
   where $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$.

Discussion. We would like to highlight the following aspects of the above definition:

- **Maliciously generated CRS^*:** In the above definition, the adversary is the one who is choosing the CRS^* which might be maliciously generated. Naturally, our aim is to capture the case where the maliciously generated CRS^* is indistinguishable from an honestly generated one, but the malicious authority has some trapdoors that enable it to extract sensitive information. However, the definition also has to capture the case where CRS^* might be far from an honestly generated CRS, to the extent where the scheme does not even provide correctness with respect to that CRS^*. In that case, the output of the real experiment is 0, and it is easy to implicate the adversary.

- **Oracle access to the adversary:** We remark that the extractor only has an oracle access to the authority and does not get the code of the authority. Giving the code of the authority to the extractor is unrealistic - in real life, the authority is an actual entity, and so the extractor will not have access to its code. Moreover, we want to explicitly prevent the extractor from "rewinding" the authority, as this is not realistic in the real world.
- **The distribution of the queries of the extractor:** The malicious authority $\mathcal{A}$ is the same in both experiments, and as such its view, as generated by the extractor, should be indistinguishable from the view in the real execution. In particular, this means that the malicious authority should receive from $\mathcal{E}$ in each iteration a pair (x_i, π_i) where the marginal distribution on x_i is computationally indistinguishable from the marginal distribution on x_i sampled according to the distribution $(x_i, w_i) \leftarrow \mathcal{D}$ and then setting $\pi_i = \mathsf{Prove}(\mathsf{CRS}^*, x_i, w_i)$.

4.2 Malicious Authority for Secure Two-Party Computation

We now extend the above definition to general two-party computation in the CRS model. We again assume that the reader is familiar with the standard (standalone) definition of secure computation, and refer to the full version for a formal definition. We require that the protocol Π simulates some functionality $\mathcal{F}$ in the CRS model. Then, we add the malicious authority capability. We first provide the definition and then discuss its changes from the NIZK definition. We define a judgement algorithm similarly to the NIZK case:

- $b \leftarrow \mathsf{Judge}(\mathsf{CRS}, \tau)$: It takes as input a common reference string CRS, a certificate τ and outputs a bit b, indicating whether the common reference string CRS is corrupted or not.

Definition 4.2. *We say that a protocol $\Pi = (\pi, \mathsf{Judge})$ has malicious authority security for the functionality $\mathcal{F}$ with respect to the distribution $\mathcal{D}$ if the following conditions hold:*

1. **Simulation security:** *π satisfies simulation security for the functionality $\mathcal{F}$.*
2. **Accountability:** *We say that π satisfies* security against malicious authority *with respect to the distribution $\mathcal{D}$ if the same conditions as in Definition 4.1 hold, where now the random variables $\mathsf{Acc.Real}_{\pi,\mathcal{A},q}(\lambda)$ and $\mathsf{Acc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)$ are defined as follows.*

   $\mathsf{Acc.Real}_{\pi,\mathcal{A},q}(\lambda)$*:*

   (a) $\mathcal{A}(1^\lambda)$ is invoked and outputs a common reference string CRS^*.

   (b) The protocol π is executed q number of times, where in the kth execution:
   - *The adversary $\mathcal{A}$ chooses some input $x_i^{(k)}$ and randomness $r_i^{(k)}$ for P_i.*
   - *The input $x_j^{(k)}$ of the honest party is sampled according to $\mathcal{D}$.*

- *The protocol is run on these inputs; let* trans_k *be the resulting transcript.*
- *The adversary* $\mathcal{A}$ *receives* trans_k*.*

(c) The adversary outputs $(k, x_j^{(k)})$ *for some* $k \in [q]$*.*

(d) The output of the experiment is 1 *if the input of* P_j *in the* k*th execution was* $x_j^{(k)}$*.*

$\underline{\mathsf{Acc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)}$*:*

(a) $\mathcal{A}(1^\lambda)$ *is invoked and output common reference string* CRS^**.*

(b) The following is run for q *number of times:*

- *The adversary outputs some input* $x_i^{(k)}$ *and randomness* $r_i^{(k)}$ *for* P_i*.*
- *The extractor replies with* trans_k*.*

(c) The $\mathcal{A}$ *receives the* q *queries, it outputs* $\left(k, x_j^{(k)}\right)$ *for some* $k \in [q]$*. The extractor* $\mathcal{E}$ *then outputs* τ*.*

(d) The experiment outputs 1 if $\mathsf{Judge}(\mathsf{CRS}^*, \tau) = \mathsf{corrupted}$*.*

3. **Defamation free:** *Same as in Definition 4.1.*

Comparison to the malicious authority security of NIZK. Malicious authority security of NIZK is a special case of the above definition for two-party computation. In NIZK, the functionality involves a prover and a verifier, where the prover sends the functionality some (x, w). If $(x, w) \in R_L$ then the functionality sends (x, yes) to the verifier and otherwise it sends (x, no). The protocol in the CRS model consists of a single message from the Prover to the Verifier. As we are interested in the privacy of the witness, we focus on the case where the verifier is corrupted. As a result, the input of the honest party (the prover) is chosen according to the distribution $\mathcal{D}$ and the input of the corrupted party (the verifier) is chosen by the adversary, which is empty in the case of NIZK. The adversary receives the transcript and at some point has to come up with the input of the honest party, i.e., extract from (x_i, π_i) some (x_i, w_i).

4.3 Strong Accountability

So far, we considered adversaries that are passive - while they contribute the input and randomness of the corrupted party in the protocol, the malicious authority expects to get back a full transcript of the protocol, i.e., the adversary is semi-malicious. Here, we model the case where the malicious authority is part of the protocol and colludes with one of the parties. Recall that simulation security is guaranteed only if the CRS authority honestly generated the CRS. If it does not, then we just have accountability as defined next. Defamation free is defined similarly to the previous definitions.

Definition 4.3. *We say that* π *satisfies* strong security against malicious authority *with respect to the distribution* $\mathcal{D}$ *if the conditions as in Definition 4.1 hold, where now the random variables* $\mathsf{StrongAcc.Real}_{\pi,\mathcal{A},q}(\lambda)$ *and*

$\mathsf{StrongAcc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)$ *are defined as follows.*

$\underline{\mathsf{StrongAcc.Real}_{\pi,\mathcal{A},q}(\lambda)}$*:*

1. $\mathcal{A}(1^\lambda)$ *is invoked and outputs a common reference string* CRS^*.
2. *The protocol* π *is executed* q *number of times where in the* k*th execution:*
   - *The adversary* $\mathcal{A}$ *participates in the protocol and corrupts party* P_i.
   - *The input* $x_j^{(k)}$ *of the honest party is sampled according to* $\mathcal{D}$.
3. *The adversary outputs* $(k, x_j^{(k)})$ *for some* $k \in [q]$ *and* $j \neq i$.
4. *The output of the experiment is* 1 *if the input of* P_j *in the* k*th execution was* $x_j^{(k)}$.

$\underline{\mathsf{StrongAcc.Ext}_{\pi,\mathcal{A},\mathcal{E},q}(\lambda)}$*:*

1. $\mathcal{A}(1^\lambda)$ *is invoked and outputs a common reference string* CRS^*.
2. $\mathcal{A}$ *and* $\mathcal{E}$ *engage in* q *executions of a secure computation protocol, where the party* P_i *is controlled by* $\mathcal{A}$.
3. *After the* q *executions,* $\mathcal{A}$ *outputs* $(k, x_j^{(k)})$ *for some* $k \in [q]$ *and* $j \neq i$. *The extractor* $\mathcal{E}$ *then outputs* τ.
4. *The experiment outputs 1 if* $\mathsf{Judge}(\mathsf{CRS}^*, \tau) = \mathsf{corrupted}$.

5 Malicious Authority Security for NIZK

In this section we construct a NIZK satisfying malicious authority security for the language of circuit satisfiability on committed inputs. The instance is a circuit $C : \{0,1\}^m \to \{0,1\}$, some committed values $\mathbf{c} = (c_1, \ldots, c_m)$ and the output b. The claim is that $\mathbf{c}$ are commitments of bits $\mathbf{x} = (x_1, \ldots, x_m)$, and that $C(x_1, \ldots, x_m) = b$. Formally:

$$R_{\mathsf{p}}(L) = \Big\{(C, c_1, \ldots, c_m, b) \mid \exists (x_1, \ldots, x_m) \in \{0,1\}^m, \\ (r_1, \ldots, r_m) \in \{0,1\}^{\mathsf{poly}(\lambda)} \text{ s.t. } C(x_1, \ldots, x_m) = b \\ \wedge \ \forall i \in [m] \ c_i = \mathsf{Com}(\mathsf{p}, x_i; r_i)\Big\}$$

The public parameters p associated with the commitment scheme are part of the CRS. We let $\mathcal{C}$ be the set of all circuits that map m-bit input to 1 bit output.

Tools. The construction is based on the following components:

- A pseudorandom generator $\mathsf{PRG} : \{0,1\}^\lambda \to \{0,1\}^{\mathsf{poly}(\lambda)}$.
- A rerandomizable bit commitment scheme which is perfectly binding and computationally hiding, denoted as $\mathsf{C} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Rerand})$, where $\mathsf{Setup}(1^\lambda)$ outputs some public parameters p, $\mathbf{c} \leftarrow \mathsf{Com}(\mathsf{p}, m; r)$ where m is a bit and r is the de-commitment. $\mathbf{c}' \leftarrow \mathsf{Rerand}(\mathsf{p}, \mathbf{c}; s)$ takes a commitment $\mathbf{c}$ and randomness s and outputs some commitment $\mathbf{c}'$.

It holds that $\mathsf{Rerand}(\mathsf{p}, \mathsf{Com}(\mathsf{p}, m; r); s) = \mathsf{Com}(\mathsf{p}, m, s')$ for some $s' = f_{\mathsf{com}}(r, s)$, and for every fixed s, r, the two functions $f_{\mathsf{com}}(\cdot, s)$ and $f_{\mathsf{com}}(r, \cdot)$ are bijections.

– Two non-interactive witness-indistinguishable proof systems $(\mathsf{Prove}, \mathsf{Verify})$, denoted as $\Pi^{(1)}_{\mathsf{NIWI}}, \Pi^{(2)}_{\mathsf{NIWI}}$, associated with the languages L_1 and L_2, respectively, which are defined as follows:

- **The language L_1:**

$$\textbf{Statement}: \ \left(\mathsf{p}, (c_0^0, c_1^0), (c_0^1, c_1^1)\right)$$
$$\textbf{Witness}: \ \left((r, \mathbf{r}_0^0, \mathbf{r}_1^0), (r, \mathbf{r}_0^1, \mathbf{r}_1^1)\right) \ ,$$

such that *one* of the following conditions hold:

1. $\mathsf{p} = \mathsf{Setup}(1^\lambda; r)$ and $(c_0^0, c_1^0) = (\mathsf{Com}(\mathsf{p}, 0, \mathbf{r}_0^0), \mathsf{Com}(\mathsf{p}, 1, \mathbf{r}_1^0))$, or
2. $\mathsf{p} = \mathsf{Setup}(1^\lambda; r)$ and $(c_0^1, c_1^1) = (\mathsf{Com}(\mathsf{p}, 0, \mathbf{r}_0^1), \mathsf{Com}(\mathsf{p}, 1, \mathbf{r}_1^1))$.

- **The language L_2:**

$$\textbf{Statement}: \ \left(\mathsf{CRS}, C, \{c_i\}_{i\in[m]}, b\right)$$
$$\textbf{Witness}: \ \left(\mathsf{s_{in}}, \{x_i\}_{i\in[m]}, \{r_i\}_{i\in[m]}, \sigma, \{s_i\}_{i\in[m]}\right)$$

such that *one* of the following conditions hold:

1. $\mathsf{PRG}(\mathsf{s_{in}}) = \mathsf{s_{out}}$, or,
2. $C(x_1, \ldots, x_m) = b$, and $\forall i \in [m]$ it holds that $c_i = \mathsf{Com}(\mathsf{p}, x_i; r_i)$.
3. $C(x_1, \ldots, x_m) = b$, and $\forall i \in [m]$ it holds that $c_i = \mathsf{Rerand}(\mathsf{p}, c^\sigma_{x_i}; s_i)$.

where $\mathsf{s_{out}}$ and $(c_0^0, c_1^0), (c_1^0, c_1^1)$ are taken from the CRS as given in Step 5 of GenCRS in Construction 5.1.

Construction 5.1: NIZK with Malicious Authority Security

GenCRS $\left(\mathbf{1}^{\boldsymbol{\lambda}}\right)$:

1. Compute $\mathsf{p} \leftarrow \mathsf{Setup}(1^\lambda; r)$ for a random r.
2. Sample $\mathsf{s_{out}} \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}$
3. For $\sigma = 0$ and $\sigma = 1$, do the following:
   – Compute $(c_0^\sigma, c_1^\sigma) = (\mathsf{Com}(\mathsf{p}, 0, \mathbf{r}_0^\sigma), \mathsf{Com}(\mathsf{p}, 1, \mathbf{r}_1^\sigma))$, for $\mathbf{r}_0^\sigma, \mathbf{r}_1^\sigma \leftarrow \{0,1\}^{\mathsf{poly}(\lambda)}$.
4. Compute $\pi^{(1)} \leftarrow \Pi^{(1)}_{\mathsf{NIWI}}.\mathsf{Prove}\left(\left((c_0^0, c_1^0), (c_0^1, c_1^1)\right), (r, \mathbf{r}_0^0, \mathbf{r}_1^0, \bot, \bot, \bot\right)$.
5. Output $\mathsf{CRS} = \left(\mathsf{p}, \mathsf{s_{out}}, \ (c_0^0, c_1^0), (c_0^1, c_1^1), \ \pi^{(1)}\right)$.

Prove$(\mathbf{CRS}, (\boldsymbol{C}, \boldsymbol{c_1}, \ldots, \boldsymbol{c_m}, \boldsymbol{b}), \mathbf{w})$: On input CRS, circuit $C : \{0,1\}^m \to \{0,1\}$, commitments $(c_1, \ldots, c_m)$, output bit $b \in \{0,1\}$, and witness $\mathbf{w}$ do the following:

1. Parse $\mathbf{w} = ((x_1, \ldots, x_m), (r_1, \ldots, r_m))$ where $x_i \in \{0,1\}, r_i \in \{0,1\}^{\mathsf{poly}(\lambda)}$ for all $i \in [m]$, and compute

$$\pi^{(2)} \leftarrow \Pi^{(2)}_{\mathsf{NIWI}}.\mathsf{Prove}\left((\mathsf{CRS}, C, \{c_i\}_{i\in[m]}, b), \ (\bot, \{x_i\}_{i\in[m]}, \{r_i\}_{i\in[m]}, \bot, \bot)\right) \ .$$

Output $\pi = \pi^{(2)}$.

Verify$(\mathsf{CRS}, (C, \mathbf{c}, b), \pi)$: On input CRS, instance $(C, \mathbf{c}, b)$, proof π,

1. Output the decision of $\Pi^{(2)}_{\mathsf{NIWI}}.\mathsf{Verify}((\mathsf{CRS}, C, \mathbf{c}, b), \pi)$.

Judge(CRS^*, τ) : On input (possibly maliciously generated) CRS^* and a transcript τ, do the following:

1. Check that the CRS^* is well formed. That is:
   (a) Parse $\mathsf{CRS}^* = \left(\mathsf{p}, \mathsf{s}_{\mathsf{out}},\ (c_0^0, c_1^0), (c_0^1, c_1^1),\ \pi^{(1)}\right)$.
   (b) Verify that $\Pi^{(1)}_{\mathsf{NIWI}}.\mathsf{Verify}\left(\left(\mathsf{p}, (c_0^0, c_1^0), (c_0^1, c_1^1)\right), \pi^{(1)}\right) = 1$.
   If the verification fails then abort and output $\mathsf{corrupted}$.
2. If $\tau = (\mathsf{open}, \sigma, \rho, \mathbf{r})$, where $\sigma, \rho \in \{0,1\}$ and $\mathbf{r} \in \{0,1\}^{\mathsf{poly}(\lambda)}$ then: If $c_\rho^\sigma = \mathsf{Com}(\mathsf{p}, \rho; \mathbf{r})$ then output $\mathsf{corrupted}$.
3. If $\tau = (\mathsf{rerandomize}, \sigma, \rho, \mathbf{r}, c, \mathbf{s})$ where $\sigma, \rho \in \{0,1\}$ and $\mathbf{r}, \mathbf{s} \in \{0,1\}^{\mathsf{poly}(\lambda)}$, and c is a commitment, then: If $c = \mathsf{Rerand}(\mathsf{p}, c_\rho^\sigma; \mathbf{s})$ and $c = \mathsf{Com}(\mathsf{p}, 1-\rho; \mathbf{r})$ then output $\mathsf{corrupted}$.
4. Otherwise, output honest.

The distribution $\mathcal{D}(\mathcal{D}_\mathcal{C}, \mathcal{D}_\mathbf{x})$. We define now the family of distributions $\mathcal{D}$ associated with the accountability property. We only place a restriction on the generation of commitments; that is, the commitments need to be honestly generated, i.e., the randomness r_i is uniformly distributed in $\{0,1\}^{\mathsf{poly}(\lambda)}$.

Formally, for any distribution over $\mathcal{C}$ (recall that $\mathcal{C}$ denotes the set of all circuits with m-bit input and 1-bit output) and for any distribution $\mathcal{D}_\mathbf{x}$ over $\{0,1\}^m$, the distribution $\mathcal{D}(\mathcal{D}_\mathcal{C}, \mathcal{D}_\mathbf{x})$ samples the input and witness as follows:

1. Sample a circuit C according to $\mathcal{D}_\mathcal{C}$.
2. Sample the bits $(x_1, \ldots, x_m)$ according to $\mathcal{D}_\mathbf{x}$ and evaluate $b = C(x_1, \ldots, x_m)$.
3. For every $i \in [m] \cup \{M\}$ compute $c_i = \mathsf{C.Com}(\mathsf{p}, x_i; r_i)$ for a uniform r_i.
4. Output $(C, c_1, \ldots, c_m, b)$ as the instance and $(x_1, \ldots, x_m), (r_1, \ldots, r_m)$ as the witness.

Theorem 5.2. *For every $\mathcal{D}_\mathcal{C}, \mathcal{D}_\mathbf{x}$, Construction 5.1 is a NIZK proof system with malicious authority security with respect to the distribution $\mathcal{D}(\mathcal{D}_\mathcal{C}, \mathcal{D}_\mathbf{x})$, assuming the security of* PRG, $\Pi_{\mathbf{Comm}}$, $\Pi^{(1)}_{\mathsf{NIWI}}$ *and* $\Pi^{(2)}_{\mathsf{NIWI}}$.

Proof. We show completeness, soundness, zero-knowledge, accountability and defamation-free properties.

Completeness. The completeness property follows from the completeness property of $\Pi^{(2)}_{\mathsf{NIWI}}$.

Soundness. Let $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$, and fix any PPT adversary (corrupted prover) $\mathcal{A}$ and any $(C, \mathbf{c}, b) \notin R_\mathsf{p}(L)$. We show that the probability that the adversary outputs a proof π such that $\mathsf{Verify}(\mathsf{CRS}, (C, \mathbf{c}, b), \pi) = 1$ is negligible.

We claim that $(\mathsf{s}_{\mathsf{out}}, C, \mathbf{c}, b) \notin L_2$. Once we prove this, it then follows from the perfect soundness of $\Pi^{(2)}_{\mathsf{NIWI}}$ that the probability that the adversary outputs a valid proof is negligible.

To show this, it suffices to argue that there does not exist any witness $(\mathsf{s}_{\mathsf{in}}, (x_1, \ldots, x_m), (r_1, \ldots, r_m), \sigma, \{s_i\}_{i\in[m]})$ for the instance $(\mathsf{s}_{\mathsf{out}}, C, \mathbf{c}, b)$.

1. Firstly, with overwhelming probability it holds that $\mathsf{PRG}(\mathsf{s}_{\mathsf{in}}) \neq \mathsf{s}_{\mathsf{out}}$ since $\mathsf{s}_{\mathsf{out}}$ is sampled uniformly at random.
2. Since $(C, \mathbf{c}, b) \notin R_{\mathsf{p}}(L)$, it holds that either $C(x_1, \ldots, x_m) \neq b$, or there exists some $i \in [m]$ such that $c_i \neq \mathsf{Com}(\mathsf{p}, x_i; r_i)$.
3. Finally, we argue that there exists some $i \in [m]$ such that $\mathsf{Rerand}(\mathsf{p}, c^{\sigma}_{x_i}; s_i) \neq c_i$. Observe that if c_i was obtained as a rerandomization of some $c^{\sigma}_{x_i}$, then still there exists a unique opening to x_i (or no opening at all). This would then violate $(C, \mathbf{c}, b) \notin R_{\mathsf{p}}(L)$.

Zero-Knowledge. We describe a PPT simulator Sim. The simulator Sim runs GenCRS, but compute s_{out} as an output of PRG where the seed s_{in} is sampled uniformly at random from $\{0,1\}^{\lambda}$ and stores the trapdoor s_{in}. Then, whenever it is given an instance $(\mathsf{CRS}, C, \mathbf{c}, b)$ it uses $(\mathsf{s}_{\mathsf{in}}, \bot, \bot, \bot, \bot)$ as a witness to compute the proof $\pi^{(2)}_{\mathsf{NIWI}}$.

The computational indistinguishability of the real world and the ideal world follows from the witness-indistinguishability of $\Pi^{(2)}_{\mathsf{NIWI}}$, and the pesudorandomness of PRG.

Accountability. We first describe the extractor $\mathcal{E}$. On input (possibly maliciously generated) CRS^* generated by the malicious authority, it does the following:

1. It checks that CRS^* is well formed, as described in Step 1 in Judge algorithm. If CRS^* is not well formed, then it halts and outputs $\tau = \bot$.
2. It chooses a branch $\sigma \in \{0,1\}$ uniformly at random and runs the following for q iterations. For every $j \in [q]$:
   (a) It samples a circuit $C^{(j)} = C$ according to the distribution $\mathcal{D}_{\mathcal{C}}$.
   (b) It samples an input $\mathbf{x}^{(j)} = \mathbf{x} = (x_1, \ldots, x_m)$ according to the distribution $\mathcal{D}_{\mathbf{x}}$.
   (c) It evaluates $b^{(j)} = b = C(x_1, \ldots, x_m)$.
   (d) For every $i \in [m]$, it generates the commitment $\hat{c}_{x_i} = \mathsf{C.Rerand}(\mathsf{p}, c^{\sigma}_{x_i}, s_i)$ for a uniformly random $s^{(j)}_i = s_i \in \{0,1\}^{\mathsf{poly}(\lambda)}$.
   Let $\hat{\mathbf{c}}^{(j)} = \hat{\mathbf{c}} = (\hat{c}_{x_1}, \ldots, \hat{c}_{x_m})$ be the sequence of all commitments.
   (e) It generates $\pi = \pi^{(2)} \leftarrow \Pi^{(2)}_{\mathsf{NIWI}}.\mathsf{Prove}\left((\mathsf{CRS}, C, \hat{\mathbf{c}}, b), (\bot, \bot, \bot, \sigma, \{s_i\}_{i\in[m]})\right)$.
   (f) It sends $((C, \hat{\mathbf{c}}, b), \pi)$ to the malicious authority.
3. After q iterations, the malicious authority might reply with some input $(j, (C, \hat{\mathbf{c}}, b))$ with witness $\{x_i\}_{i\in[m]}$ and $\{\mathbf{r}_i\}_{i\in[m]}$ for which $\hat{c}_i = \mathsf{Com}(\mathsf{p}, x_i; \mathbf{r}_i)$ for all $i \in [m]$ and $C(x_1, \ldots, x_m) = b$.
   (a) If $(x_1, \mathbf{r}_1)$ satisfies $c^{\sigma}_{x_1} = \mathsf{Com}(\mathsf{p}, x_1; \mathbf{r}_1)$ where $c^{\sigma}_{x_1}$ is taken from the CRS, then output $\tau = (\mathsf{open}, \sigma, x_1, \mathbf{r}_1)$.

(b) Otherwise, if it holds that $c' := \mathsf{Rerand}(\mathsf{p}, c^{\sigma}_{1-x_1}, \mathbf{s}_1^{(j)}) = \mathsf{Com}(\mathsf{p}, x_1, \mathbf{r}_1)$, output $\tau = (\mathsf{rerandomize}, \sigma, x_1, \mathbf{r}, c', \mathbf{s}_1^{(j)})$.

We claim that if the extractor chooses σ to be the "correct branch", i.e., the one for which $\pi^{(1)}$ from the CRS is correct, then the view of the malicious authority is indistinguishable between $\mathsf{Acc.Real}$ and $\mathsf{Acc.Ext}$. We then claim that the extractor always outputs a transcript that is accepted by Judge if the malicious authority outputs a valid witness $(j, (C, \hat{\mathbf{c}}, b))$ for the $j \in [q]$ iteration. To show this, we need to argue that the view of the authority when interacting with the real parties is computationally indistinguishable from the view of the authority when interacting with the extractor.

Indistinguishability of views. Consider the following hybrids:

- $\mathbf{H}_1$: We change the way CRS is generated, such that $\mathsf{s}_{\mathsf{out}}$ is an output of PRG where the seed s_{in} is sampled uniformly at random from $\{0,1\}^{\lambda}$. The output of the hybrid is 1 if the adversary succeeds to output π for which $\mathsf{Verify}(\mathsf{CRS}, (C, \mathbf{c}, b), \pi) = 1$.
- $\mathbf{H}_2$: This is the experiment $\mathsf{Acc.Real}_{\Pi,\mathcal{A},q}(\lambda)$.
- $\mathbf{H}_3$: This hybrid is inefficient. The prover in the real experiment works as follows: It samples $C \leftarrow \mathcal{D}_{\mathcal{C}}$ and $\mathbf{x} \leftarrow \mathcal{D}_{\mathbf{x}}$. Then, instead of directly committing to the $\mathbf{x} = (x_1, \ldots, x_m)$ it takes $c^{\sigma}_{x_1}, \ldots, c^{\sigma}_{x_m}$. It then rerandomizes all the commitments to obtain rerandomized commitments $\hat{c}^{\sigma}_{x_1}, \ldots, \hat{c}^{\sigma}_{x_m}$. It then runs in exponential time and determines the randomness $\{\mathbf{r}_i\}_{i\in[m]}$ such that $\hat{c}^{\sigma}_{x_i} = \mathsf{Com}(\mathsf{p}, x_i; \mathbf{r}_i)$. It uses $(\bot, x, \{r_i\}_{i\in[m]}, \bot, \bot)$ to compute the proof $\pi^{(2)}_{\mathsf{NIWI}}$ for the instance $(C, \widetilde{\mathbf{c}}, b)$, where $b = C(x)$. The rest of the hybrid is the same as before.
- $\mathbf{H}_4$: This is $\mathsf{Acc.Ext}_{\Pi,\mathcal{A},q}(\lambda)$ with branch σ.

From the security of PRG, it follows that the view of the adversary in $\mathbf{H}_1$ is computational indistinguishale from the real view. The view of the adversary in $\mathbf{H}_2$ and $\mathbf{H}_3$ is identical, which follows from the fact that the rerandomization procedure of the commitment scheme generates commitments that are identically distributed to fresh commitments.

The two hybrids $\mathbf{H}_3$ and $\mathbf{H}_4$ are computationally indistinguishable, follows from the witness-indistinguishability of $\Pi^{(2)}_{\mathsf{NIWI}}$. We consider a non-uniform reduction which gets as input the CRS and the decommitments of the commitments in the CRS as non-uniform advice. Using this, the (efficient) reduction computes and sends the instance $(C, \widehat{c}, b)$, where $b = C(x)$, along with two witnesses to the challenger of the WI game. The challenger then sends the proof $\pi^{(2)}_{\mathsf{NIWI}}$ to the reduction, who forwards it to the adversary. If the adversary can distinguish the hybrids $\mathbf{H}_3$ and $\mathbf{H}_4$ with non-negligible probability then the reduction can break the witness-indistinguishability property with non-negligible probability, a contradiction.

Once the authority gives a witness $\{x_i\}_{i\in[m]}$ and $\{\mathbf{r}_i\}_{i\in[m]}$ for the jth execution, from inspection it is easy to see that the extractor outputs a transcript

that implicates the authority. Thus, if the authority succeeds in Acc.Real with some non-negligible probability $\epsilon_1(\lambda)$ then the extractor succeeds with probability negligibly close to $\epsilon_1(\lambda)/2$, where the loss occurs from guessing σ and the indistinguishability of the proof π.

Defamation-Free. For every PPT adversary $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that: $\Pr[\mathsf{Judge}(\mathsf{CRS}, \mathcal{A}(\mathsf{CRS})) = \mathsf{corrupted}] \leq \mu(\lambda)$, where $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$. First, since CRS is honestly generated, it always passes the verification of Step 1 in the Judge algorithm. We now show that for every $\sigma \in \{0,1\}$, no PPT adversary can output $\tau = (\mathsf{open}, \sigma, \cdot)$ or $\tau = \mathsf{rerandomize}, \sigma, \cdot, \cdot, \cdot, \cdot)$ with non-negligible probability. For that, fix $\sigma \in \{0,1\}$, and consider the following sequence of hybrid experiments:

1. $\mathbf{H}_1$:
   In this hybrid, the adversary receives $\mathsf{CRS} \leftarrow \mathsf{GenCRS}(1^\lambda)$ and the output of the hybrid is 1 if it outputs $\tau = (\mathsf{open}, \sigma, \cdot\cdot)$ or $\tau = (\mathsf{rerandomize}, \sigma, \cdot, \cdot, \cdot, \cdot)$ that is accepted by Judge.
2. $\mathbf{H}_2$: We modify the way CRS is generated, such that for proving the instance $((c_0^0, c_1^0), (c_0^1, c_1^1))$ using $\Pi^{(1)}_{\mathsf{NIWI}}$, we use the witness of $1-\sigma$. That is,
   (a) If $\sigma = 0$ we prove $\pi^{(1)}$ using the witness $(\bot, \bot, \bot, r, \mathbf{r}_0^1, \mathbf{r}_1^1)$.
   (b) If $\sigma = 1$ we prove $\pi^{(1)}$ using the witness $(r, \mathbf{r}_0^0, \mathbf{r}_1^0, \bot, \bot)$.

Clearly, the views of the adversary in both the experiments are computationally indistinguishable from the witness-indistinguishability property of $\Pi^{(1)}_{\mathsf{NIWI}}$. Next, we claim that the probability that the adversary outputs an accepted transcript in $\mathbf{H}_2$ is negligible. At this point, the proof $\pi^{(1)}$ is independent of the randomness used to create the commitments c_0^σ, c_1^σ. Outputting $\tau = (\mathsf{open}, \sigma, \rho, \mathbf{r})$ is equivalent to violating the hiding property of the commitment scheme. Outputting $\tau = (\mathsf{rerandomize}, \sigma, \cdot, \cdot, \cdot, \cdot)$ is impossible in case p is perfectly binding. In case p just satisfies computational hiding, coming up with the rerandomization is equivalent to violating the (computationally infeasible) "rerandomize and open" property, as we formalize in the full version. □

6 Malicious Authority Security for Oblivious Transfer

6.1 Oblivious Transfer with Weak Accountability

Due to lack of space, we just state our results and refer the reader to the full version for further details. Our construction is based on a rerandomizable oblivious transfer, which intuitively mean that a transcript of a given execution of two-round oblivious transfer can be re-randomized, i.e., look like a fresh execution on the same inputs. We show that [31] achieves this notion. We denote by $n \times \mathsf{OT}$ the functionality of n parallel instance of 1-out-of-2 bit OT.

Theorem 6.1. *Assuming one-way functions, non-interactive witness indistinguishability proof system, and two-round rerandomizable oblivious transfer for bit OT, there exists a construction of two-round $n \times \mathsf{OT}$ with (weak) malicious authority with respect to the uniform distribution over the inputs for any $n \in \Omega(\lambda)$, where λ is the security parameter.*

6.2 Oblivious Transfer with Strong Accountability

Moreover, we show that strong accountability is possible but we need stronger assumptions:

Theorem 6.2. *Assuming the security of one-way function, indistinguishability obfuscator, non-interactive witness-indistinguishability proof system, and the existence of a rerandomizable oblivious transfer, there exists a construction that achieves* **strong** *malicious authority security for the functionality* $n \times \mathsf{OT}$ *with respect to the uniform distribution over the inputs for any* $n \in \Omega(\lambda)$*, where* λ *is the security parameter.*

7 Malicious Authority Secure for Secure 2PC

In this section, we investigate malicious authority security for secure two party computation. Due to lack of space, we just give the statements of the results and refer the reader to the full version for more details.

Lemma 7.1. *There exists a two-party functionality* F *such that for any secure computation protocol for* F *between parties* P_1 *and* P_2*, the following events cannot simultaneously hold:*

- P_i*, for some* $i \in \{1, 2\}$ *receives the output of the protocol and,*
- *The following properties are satisfied: (i) defamation-free property and, (ii) accountability holds when the malicious authority corrupts the party* P_i*.*

Positive result - the class of functions. For $\ell > \lambda$, we let $\mathcal{F}$ be a family of all functions over $F : \{0,1\}^\ell \times \mathcal{Y} \to \{0,1\}^\ell$ such that

$$F\left((x_1, \ldots, x_\ell), (y_1, \ldots, y_\ell)\right) = g\left(\{x_i\}_{y_i=1}\right) \ ,$$

for some function g. Namely, whenever $y_i = 0$, then the x_i does not affect the output. The set $\mathcal{Y} \subset \{0,1\}^\ell$ contains all elements with hamming weight at most $\ell - \lambda$.

Theorem 7.2. *For every function* $F \in \mathcal{F}$*, there exists a construction that achieves (weak) malicious authority security with respect to the uniform distribution over the inputs, assuming the existence of maliciously secure rerandomizable oblivious transfer in the CRS model, non-interactive witness-indistinguishability proofs, and pseudorandom generator.*

Acknowledgements. The authors thank the anonymous reviewers of EUROCRYPT 2021 for many helpful comments.

Gilad Asharov is sponsored by the Israel Science Foundation (grant No. 2439/20), and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in

the Prime Minister's Office. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 891234. Hila Dahari is a fellow of the Ariane de Rothschild Women Doctoral Program and supported in part by grants from the Israel Science Foundation (No. 950/15 and 2686/20) and by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness. Vipul Goyal is supported in part by the NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award.

References

1. Ananth, P., Choudhuri, A.R., Jain, A.: A new approach to round-optimal secure multiparty computation. In: Annual International Cryptology Conference. pp. 468–499. Springer (2017)
2. Ananth, P., Deshpande, A., Kalai, Y.T., Lysyanskaya, A.: Fully homomorphic NIZK and NIWI proofs. In: Theory of Cryptography TCC 2019. vol. 11892, pp. 356–385. Springer (2019)
3. Asharov, G., Orlandi, C.: Calling out cheaters: Covert security with public verifiability. In: Advances in Cryptology - ASIACRYPT 2012. vol. 7658, pp. 681–698. Springer (2012)
4. Badrinarayanan, S., Goyal, V., Jain, A., Kalai, Y.T., Khurana, D., Sahai, A.: Promise zero knowledge and its applications to round optimal mpc. In: Annual International Cryptology Conference. pp. 459–487. Springer (2018)
5. Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (im) possibility of obfuscating programs. In: CRYPTO. pp. 1–18. Springer (2001)
6. Barak, B., Ong, S.J., Vadhan, S.P.: Derandomization in cryptography. In: Advances in Cryptology - CRYPTO 2003. vol. 2729, pp. 299–315. Springer (2003)
7. Bellare, M., Fuchsbauer, G., Scafuro, A.: Nizks with an untrusted CRS: security in the face of parameter subversion. In: Advances in Cryptology - ASIACRYPT 2016. vol. 10032, pp. 777–804 (2016)
8. Benhamouda, F., Lin, H.: k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 500–532. Springer (2018)
9. Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: Symposium on Theory of Computing (STOC). pp. 103–112. ACM (1988)
10. Brakerski, Z., Halevi, S., Polychroniadou, A.: Four round secure computation without setup. In: Theory of Cryptography Conference. pp. 645–677. Springer (2017)
11. Canetti, R., Fischlin, M.: Universally composable commitments. In: Annual International Cryptology Conference. pp. 19–40. Springer (2001)
12. Canetti, R., Kushilevitz, E., Lindell, Y.: On the limitations of universally composable two-party computation without set-up assumptions. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 68–86. Springer (2003)
13. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: ACM symposium on Theory of computing (STOC). pp. 494–503 (2002)

14. Chor, B., Fiat, A., Naor, M.: Tracing traitors. In: Advances in Cryptology - CRYPTO '94. vol. 839, pp. 257–270. Springer (1994)
15. Choudhuri, A.R., Ciampi, M., Goyal, V., Jain, A., Ostrovsky, R.: Round optimal secure multiparty computation from minimal assumptions. In: Theory of Cryptography - TCC '20 (2020), to appear
16. Dwork, C., Naor, M.: Zaps and their applications. In: Foundations of Computer Science, FOCS 2000. pp. 283–293 (2000)
17. Feige, U., Lapidot, D., Shamir, A.: Multiple noninteractive zero knowledge proofs under general assumptions. vol. 29, pp. 1–28 (1999)
18. Fuchsbauer, G.: Subversion-zero-knowledge snarks. In: Public-Key Cryptography - PKC 2018. vol. 10769, pp. 315–347. Springer (2018)
19. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. SIAM Journal on Computing **45**(3), 882–929 (2016)
20. Garg, S., Goyal, V., Jain, A., Sahai, A.: Bringing people of different beliefs together to do UC. In: Theory of Cryptography - TCC 2011. vol. 6597, pp. 311–328. Springer (2011)
21. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Theory and Applications of Cryptographic Techniques (TCC). pp. 468–499 (2018)
22. Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. vol. 7, pp. 1–32 (1994)
23. Goyal, V.: Reducing trust in the PKG in identity based cryptosystems. In: CRYPTO 2007. vol. 4622, pp. 430–447. Springer (2007)
24. Goyal, V., Katz, J.: Universally composable multi-party computation with an unreliable common reference string. In: Theory of Cryptography, TCC 2008. vol. 4948, pp. 142–154. Springer (2008)
25. Groth, J., Ostrovsky, R.: Cryptography in the multi-string model. In: CRYPTO 2007. vol. 4622, pp. 323–341. Springer (2007)
26. Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive zaps and new techniques for NIZK. In: CRYPTO 2006. vol. 4117, pp. 97–111. Springer (2006)
27. Halevi, S., Hazay, C., Polychroniadou, A., Venkitasubramaniam, M.: Round-optimal secure multi-party computation. In: Annual International Cryptology Conference. pp. 488–520. Springer (2018)
28. Katz, J., Ostrovsky, R.: Round-optimal secure two-party computation. In: Annual International Cryptology Conference. pp. 335–354. Springer (2004)
29. Mukherjee, P., Wichs, D.: Two round multiparty computation via multi-key FHE. In: Advances in Cryptology - EUROCRYPT 2016. vol. 9666, pp. 735–763. Springer (2016)
30. Pass, R.: Simulation in quasi-polynomial time, and its application to protocol composition. In: Advances in Cryptology - EUROCRYPT 2003. vol. 2656, pp. 160–176. Springer (2003)
31. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Advances in Cryptology - CRYPTO 2008. vol. 5157, pp. 554–571. Springer (2008)