Towards Accountability in CRS Generation

Prabhanjan Ananth
UCSB

Gilad Asharov
Bar-Ilan University

Hila Dahari
Weizmann Institute of Science

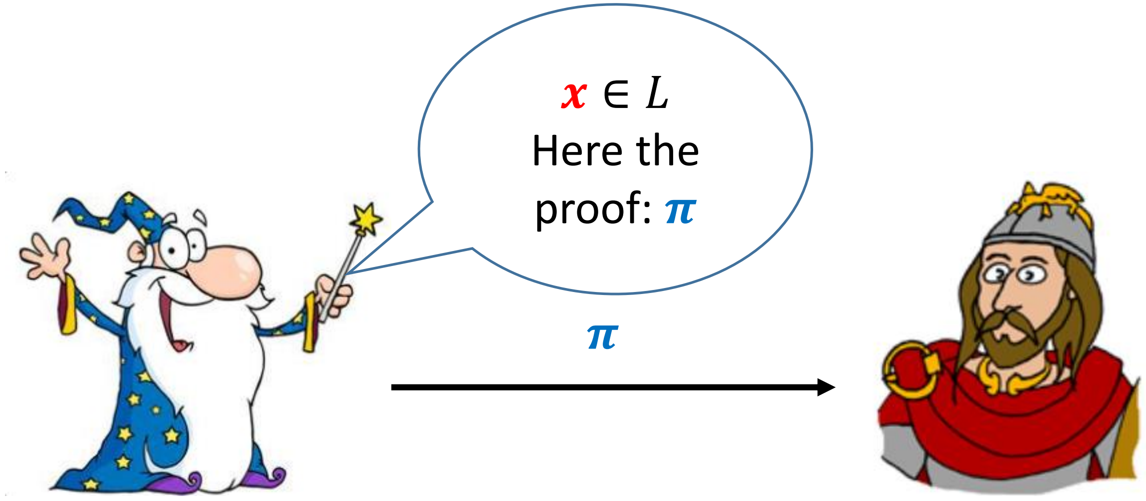
Vipul Goyal
CMU and NTT Research

Eurocrypt 2021

Non-Interactive Zero-Knowledge (NIZK) [BFM88]

The model:

- Let $\boldsymbol{L}$ be an NP-language
- Given $\boldsymbol{x}$, the **prover** wants to convince the **verifier** that $\boldsymbol{x}$ in $\boldsymbol{L}$ without revealing any **additional information** about $\boldsymbol{x}$. [GMR85]



Prover

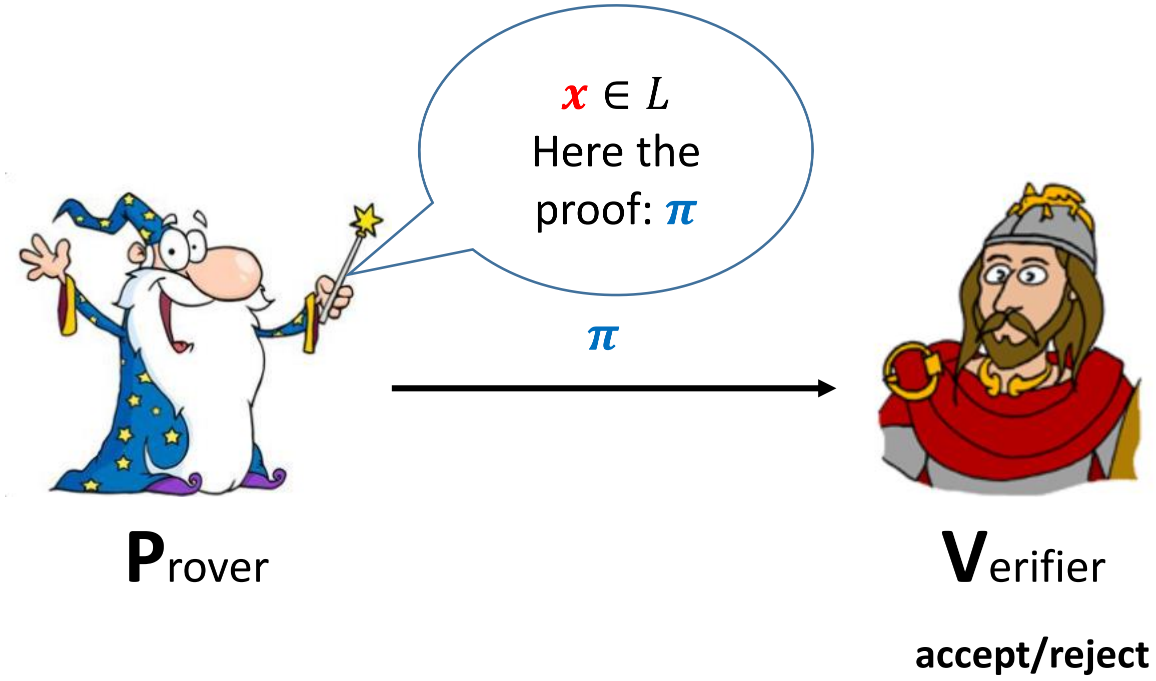
Verifier

accept/reject

Non-Interactive Zero-Knowledge (NIZK) [BFM88]

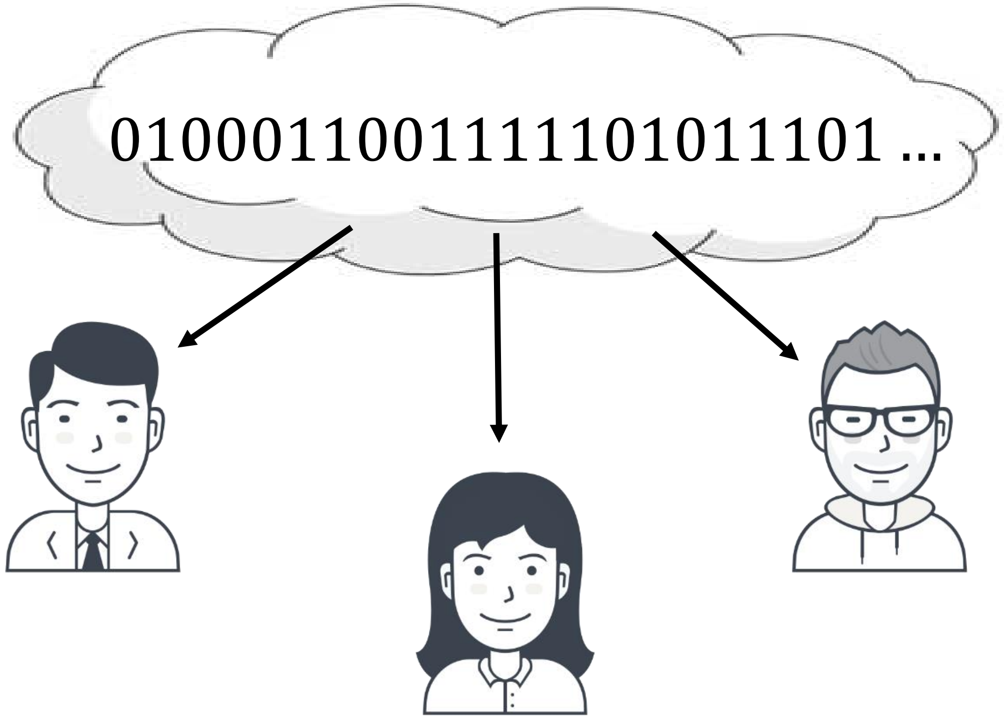
The model:

- For a **single** message zero-knowledge proof, we require **trusted set-up**, specifically, we require a **common reference string**. [GO94, FLS90]



Common Reference String (CRS) Model [BFM88,D00,FF00]

The model: The parties share a **trusted** **public string** from a known distribution.



Motivation:

- Non-interactive zero-knowledge for NP **[GO94, FLS90]**
- Malicious two round MPC **[MW16, GS18, BL18]**

Non-Interactive Zero-Knowledge (NIZK) [BFM88]

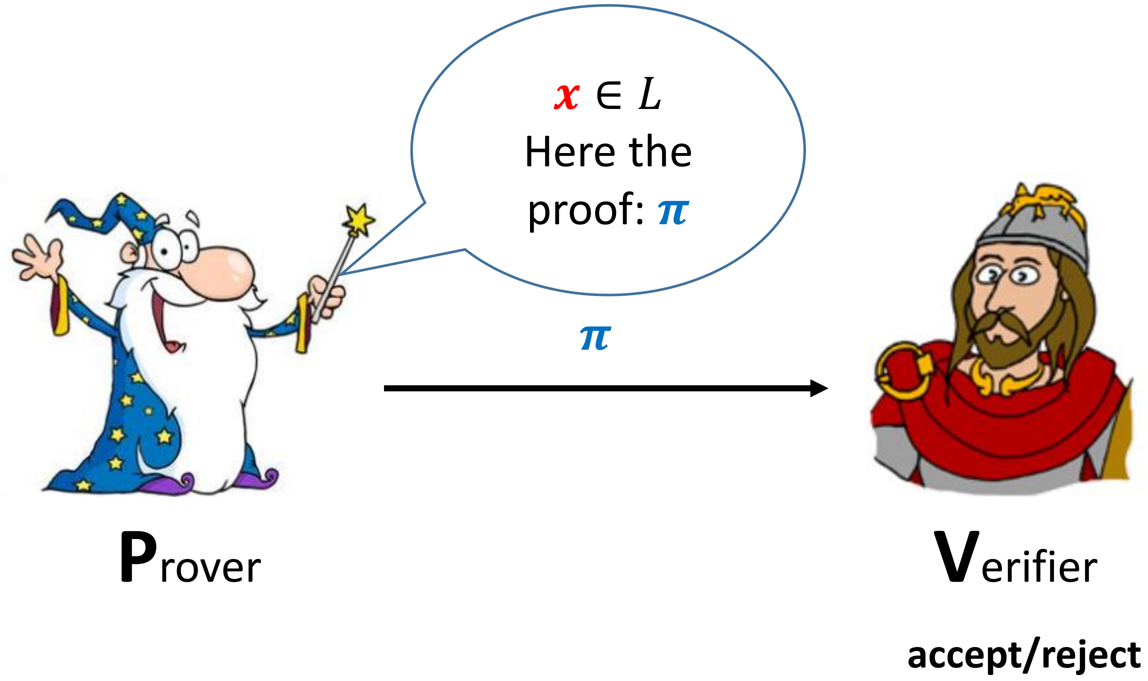
Completeness: If $x \in L$, the verifier **accepts** w.h.p

Soundness: If $x \notin L$, the verifier **rejects** w.h.p

Zero knowledge: If $x \in L$, the verifier **cannot** learn any **additional information** from the proof π.

More formally, $\exists S$ such that for all $x \in L$:

$$S(x) \cong (CRS, \pi)$$

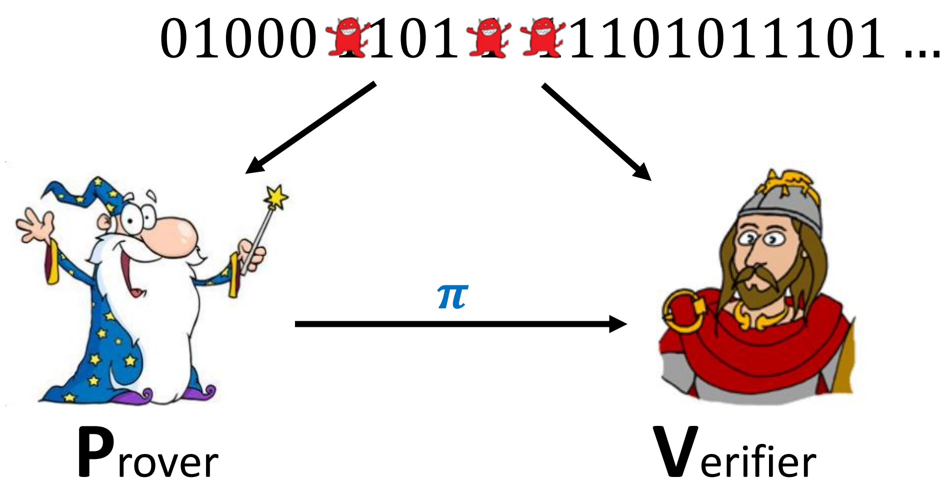


NIZK in the Common Reference String (CRS) [FLS90]

However, in the **real** world,

1. Who generates the **CRS**?

2. What happens if the **CRS** is **maliciously** generated?



Related Works

Weaker notions of security:

- Zap [DworkNaor00]
- Super-polynomial simulation security [Pas03]
- Multi-string model [GrothOstrovsky07]
- Unreliable CRS [GoyalKatz08, GargGoyalJainSahai11]
- NIZKs with an untrusted CRS [BellareFuchsbauerScafuro16]

CRS generation in the real world

Who generates the CRS?

❖ **MPC –** multiple parties generate together the CRS.

02 Dec 2016 | 18:50 GMT

The Crazy Security Behind the Birth of Zcash, the Inside Story

Zcash, the new anonymous cryptocurrency, was born in a cloak-and-dagger cocoon of digital secrecy. There was just one little problem

By **Morgen E. Peck**

Photo: Morgen Peck

Paranoia, the destroyer: Za Wilcox, brother of Zcash CEO Zooko Wilcox, sets about destroying a computer used to generate the cryptographic parameters needed to start Zcash

"How would you feel about donating your phone to science?"

Paranoia, the destroyer: Za Wilcox, brother of Zcash CEO Zooko Wilcox, sets about **destroying a computer used to generate the cryptographic parameters** needed to start Zcash

https://www.youtube.com/watch?v=D6dY-3x3teM

CRS generation in the real world

Who generates the CRS?

❖ **A trusted party**

In real life, do there really exist *trusted parties*?

CRS generation in the real world

- If a **malicious** party **recovers private** information, but **keeps** it to themselves – **impossible** to **protect** against
- If the **malicious** party **uses** the **private** information, we want to **prove** they acted maliciously

Our Talk

- Our focus: a party who tries to **sell private** information is **held accountable**
- We **introduce** the notion of **accountability** in **CRS generation**
- We study **accountability** for **NIZK**, **2PC**, and specifically, **OT**

Our Results: Informally,

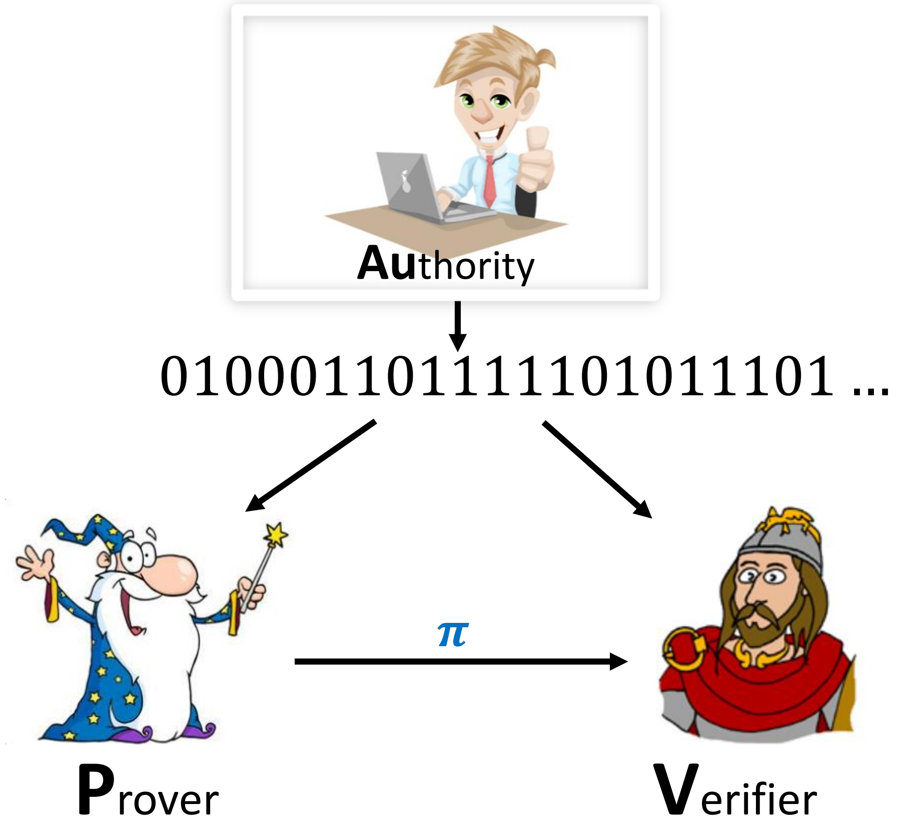
  - **NIZK:** Under standard assumptions, we get **NIZK** for **all of NP** with **accountability** in CRS generation
  - **2PC:** There is a two-party **functionality** for which it is **impossible** to achieve **accountability**
  - **2PC:** Under standard assumptions, we get **2PC** for a **large class** of **functionalities** with **accountability** in CRS generation

CRS generation in the real world

Our setting: A party called **Authority** generates the **CRS**.

- The **authority** is an **honest** party –

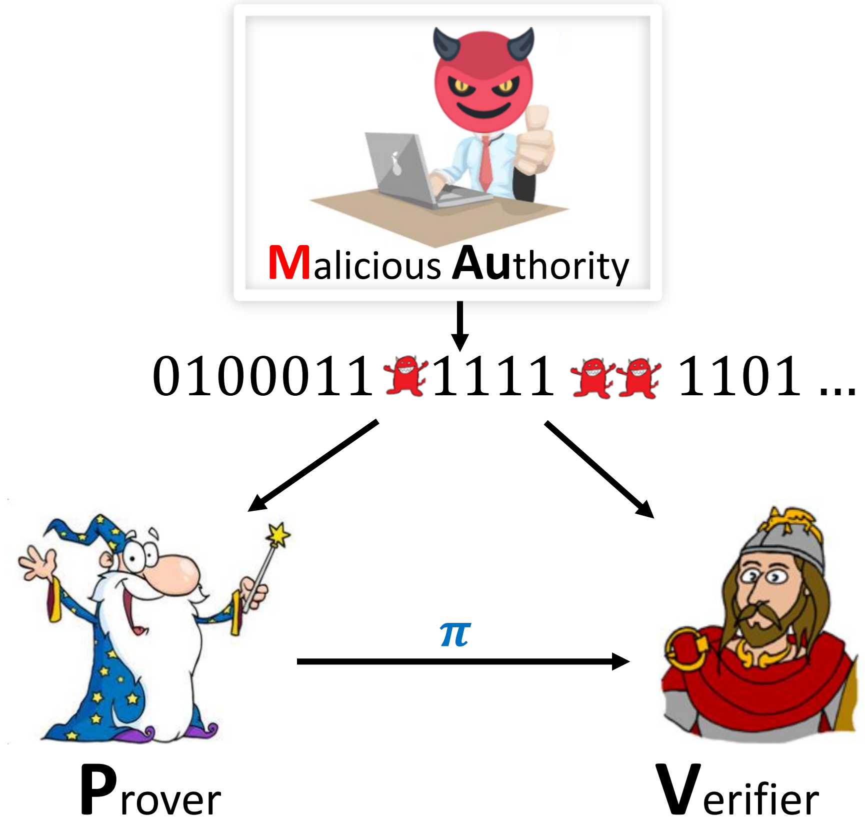
  Everything works



CRS generation in the real world

Our setting: A party called **Authority** generates the **CRS**.

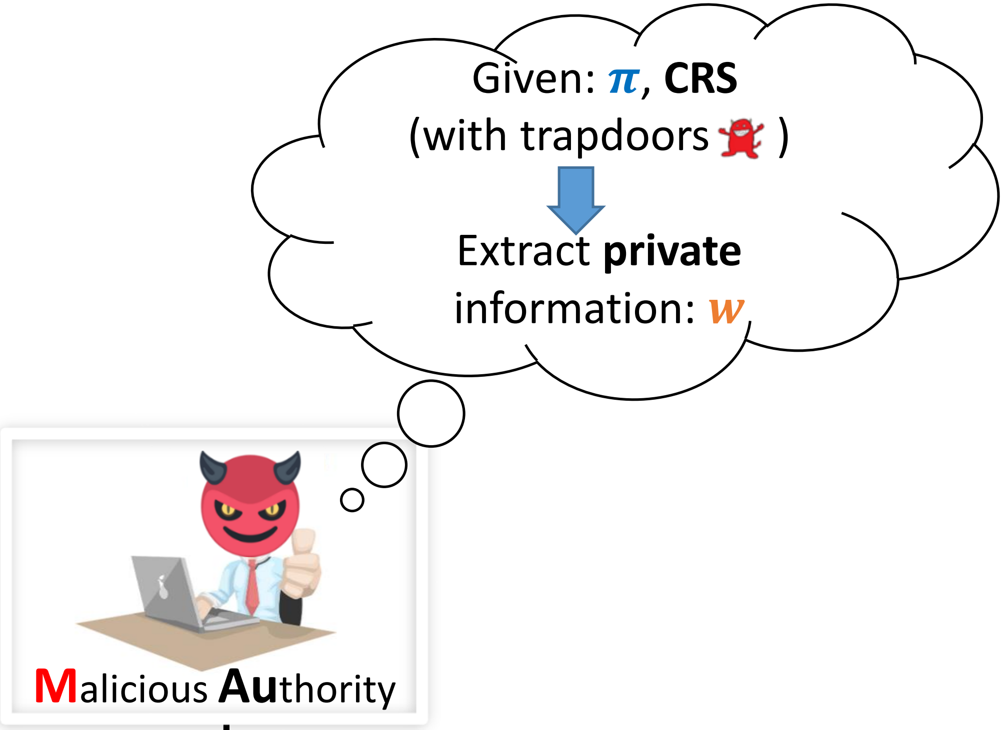
- The **authority** is a **malicious** party –
  - A malicious **authority** generates **CRS** with **trapdoors**.
  - The **prover** uses the “bad” **CRS** to generate a **NIZK** and send it to the **verifier**



CRS generation in the real world

Our setting: A party called **Authority** generates the **CRS**.

- The **authority** is a **malicious** party –
  - The malicious **authority** extracts from the proof π (using the trapdoors in the **CRS**) the **private** information w

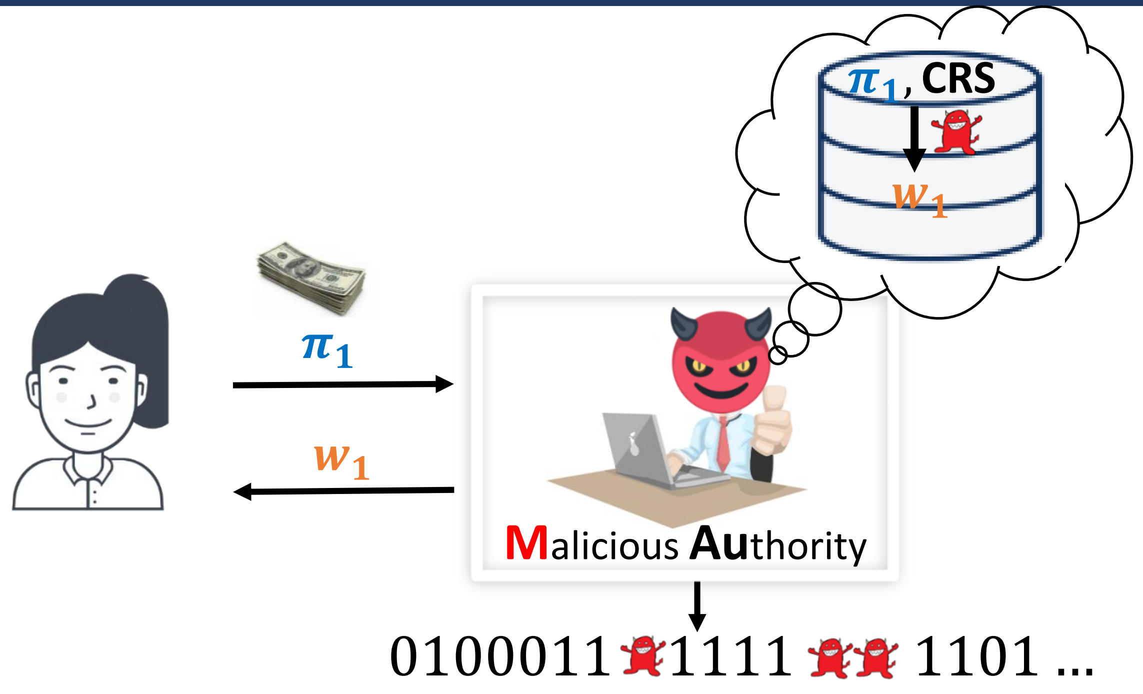


0100011 1111 1101 ...

CRS generation in the real world

Our setting: A party called **Authority** generates the **CRS**.

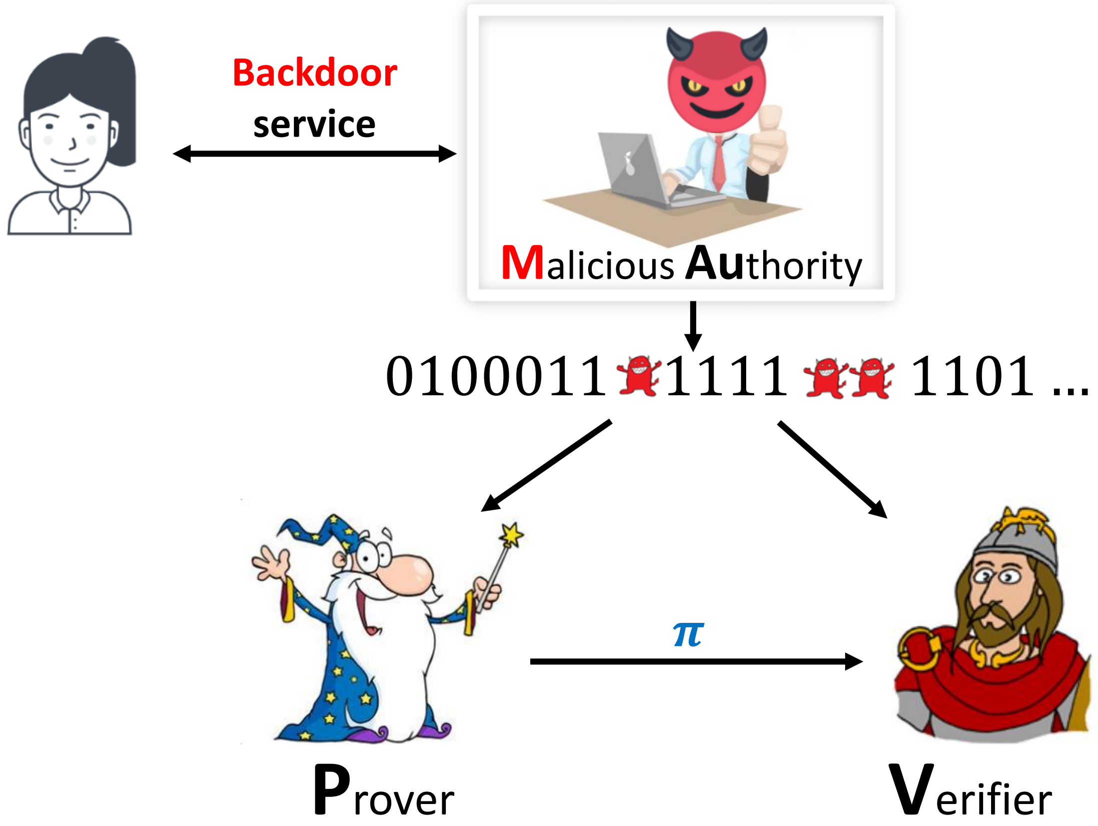
- The **authority** is a **malicious** party –
  - The malicious **authority** sets up a **backdoor** service that **sells** the **private** information w for profit



CRS generation in the real world

❖ The **authority** is a **malicious** party –

The authority can **maliciously** generate the **CRS**, with trapdoors, recover **private** information, and use the **backdoor** service to **sell** the **private** information for profit.

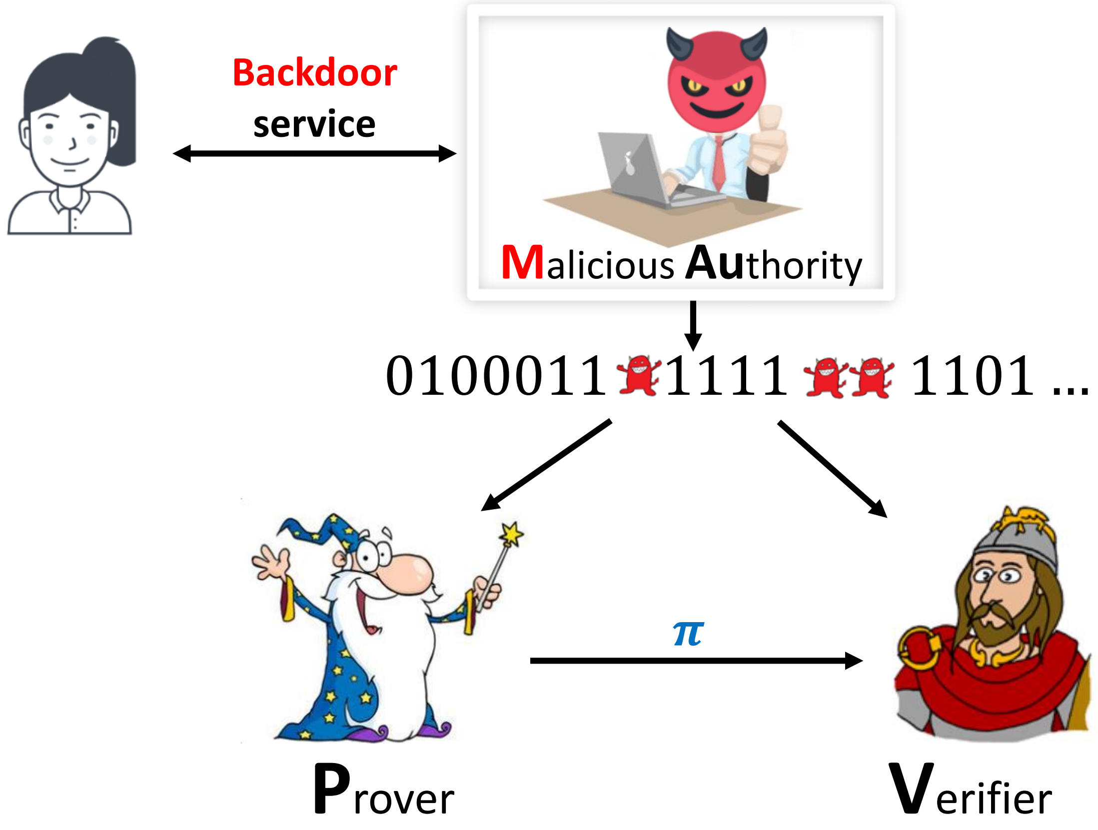


CRS generation in the real world

Our goal: Be able to use the **backdoor** service to generate a **proof** that:

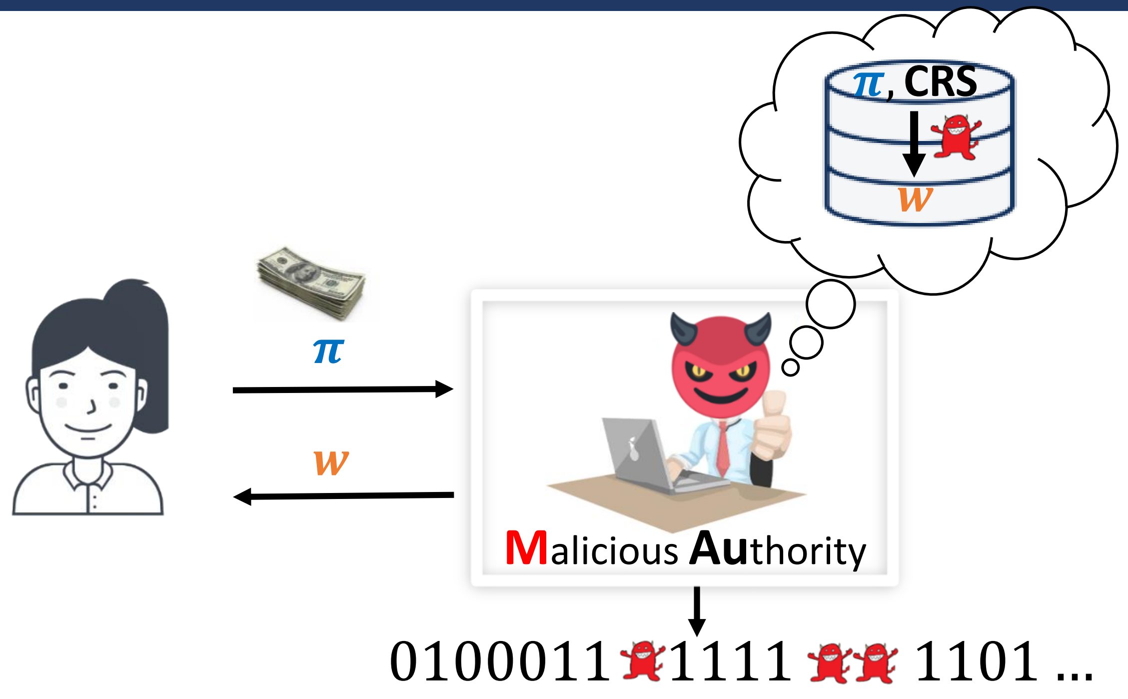
1. The **CRS** was **maliciously** generated

2. The **authority** was dishonest



CRS generation in the real world

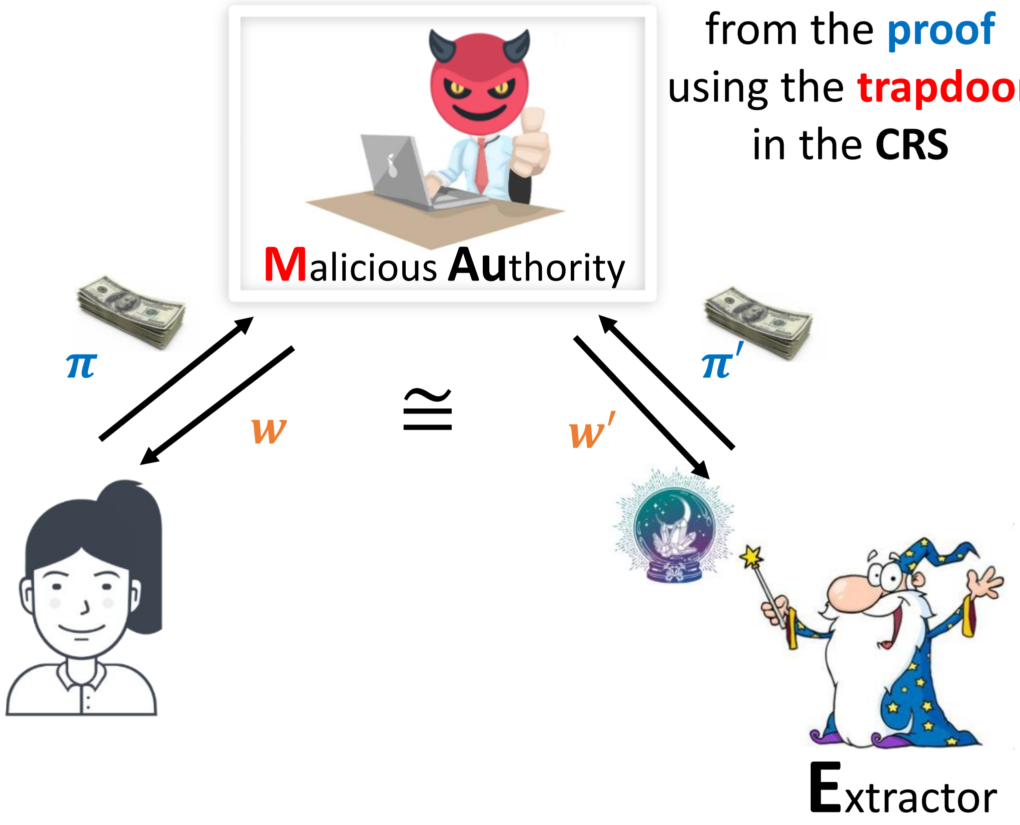
❖ Specifically, to construct an **extractor** that by **using** the **backdoor** service can generate a **proof** that the authority **maliciously** generated the **CRS**



CRS generation in the real world

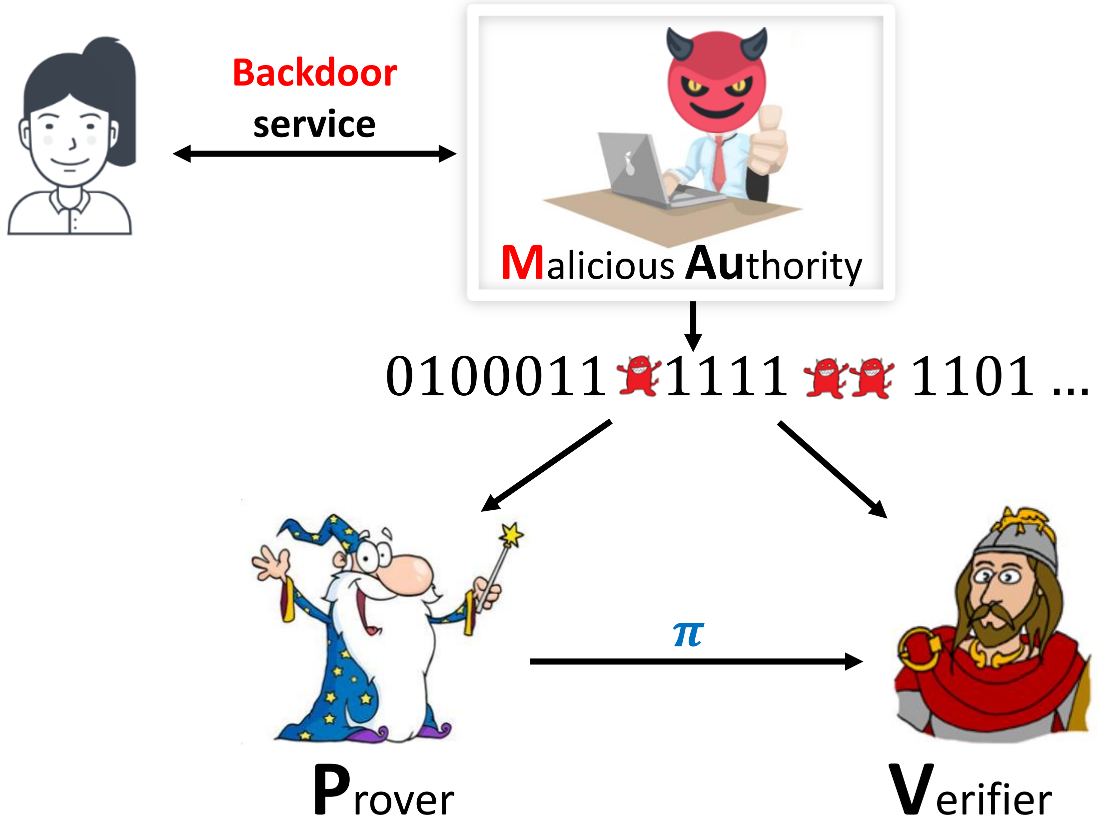
- If the **backdoor** service will recognize the **extractor**, it will **not** open the proof, thus the **queries** should look like “**real**”.

Extract the **witness** from the **proof** using the **trapdoor** in the **CRS**



CRS generation in the real world

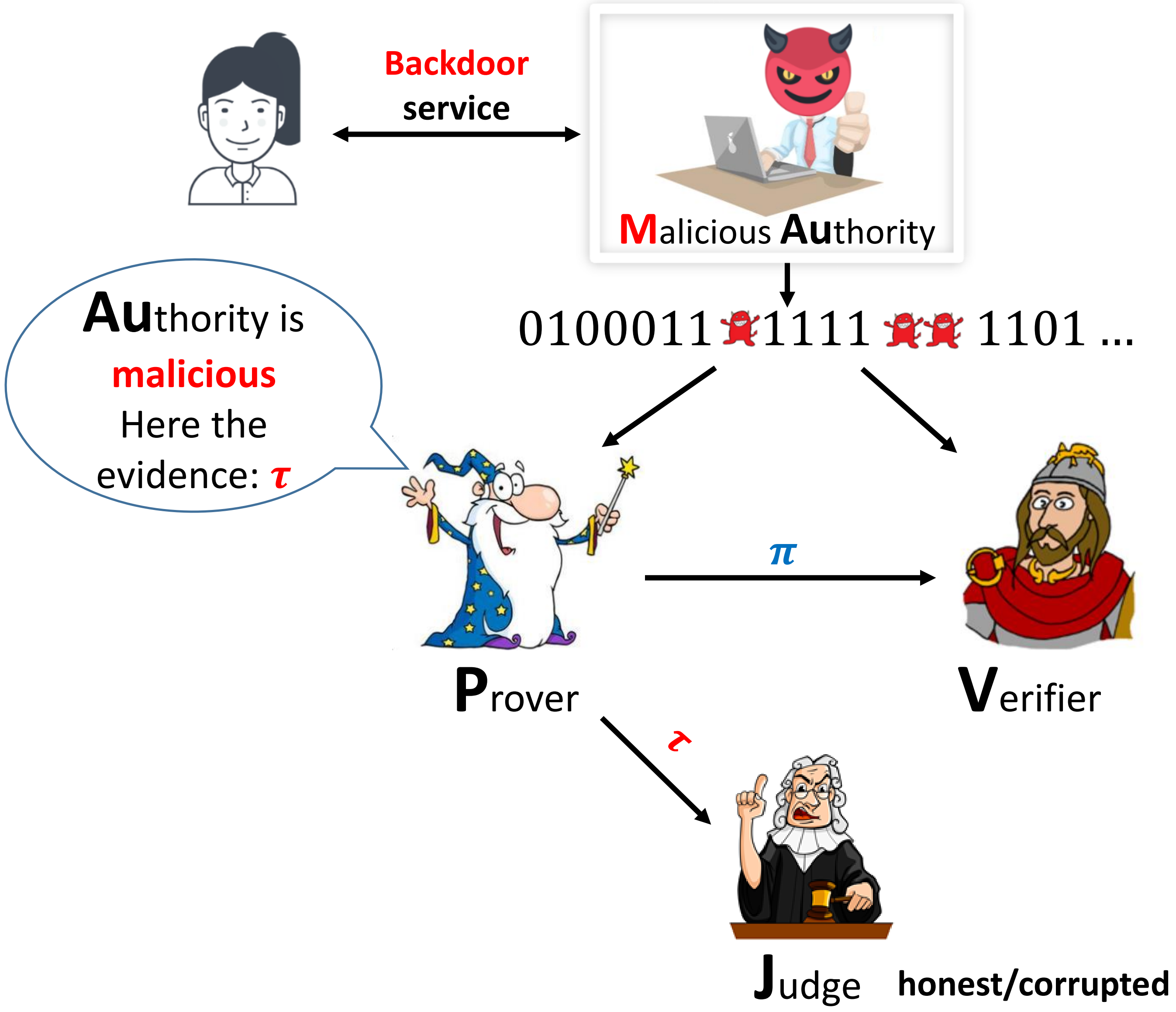
Our approach: Design a CRS generation protocol that satisfies an **accountability** property.



CRS generation in the real world

Let **(GenCRS, Prove, Verify, Judge)** be a four PPT algorithms, such that:

- **(GenCRS, Prove, Verify)** is a NIZK proof system
- **Judge** (syntax) –
  - **Input:** a **CRS**, and an evidence τ
  - **Output:** honest/corrupted **CRS**



CRS generation in the real world

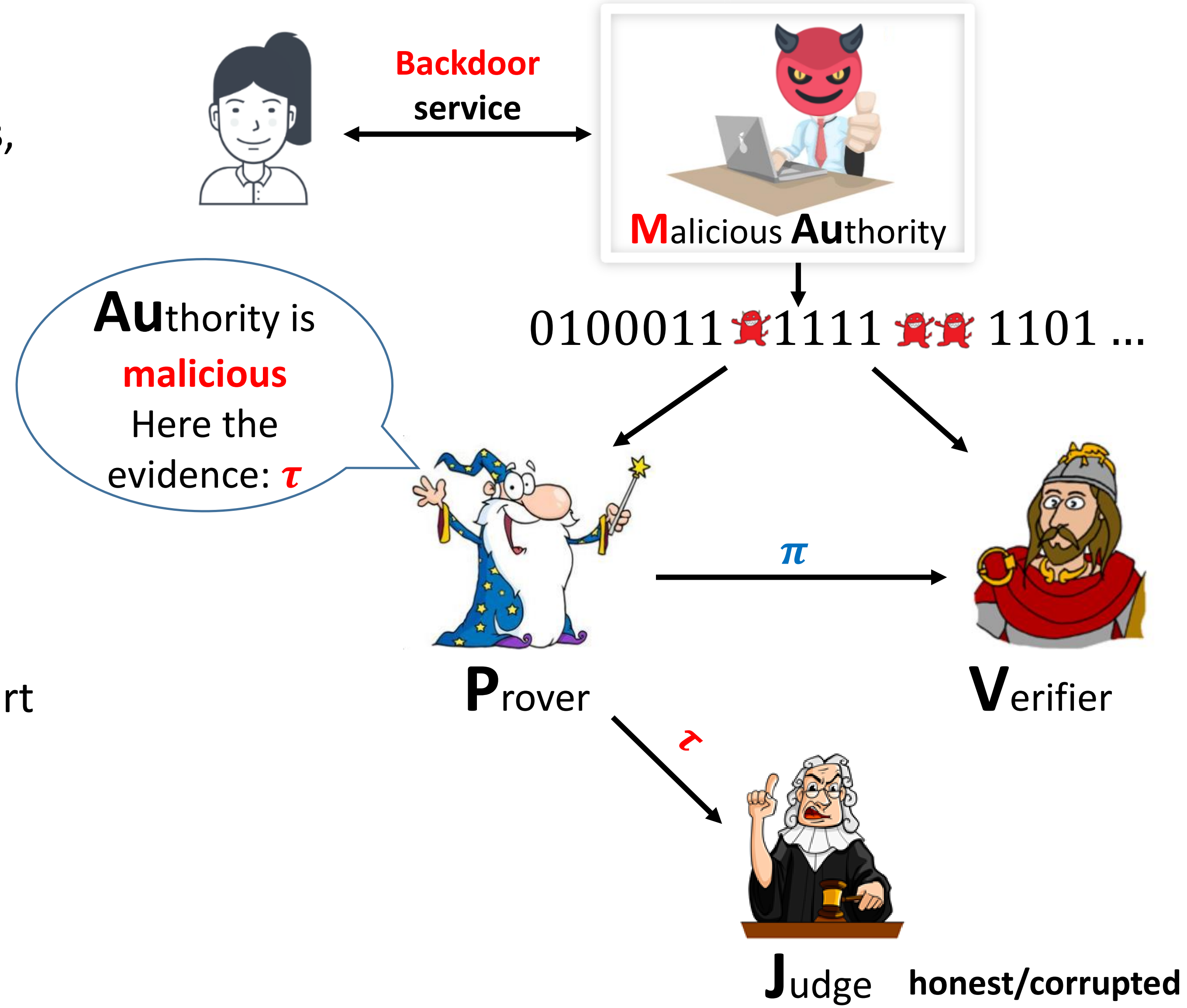
Accountability: If the **authority** is **malicious**,

and **sells** your information,

you can use the **backdoor** service to

generate a **publicly verifiable proof**.

* For example: to convince a **judge** in the court



CRS generation in the real world

Defamation free: If the **authority** is **honest**,

one **cannot** generate a **proof** against the authority

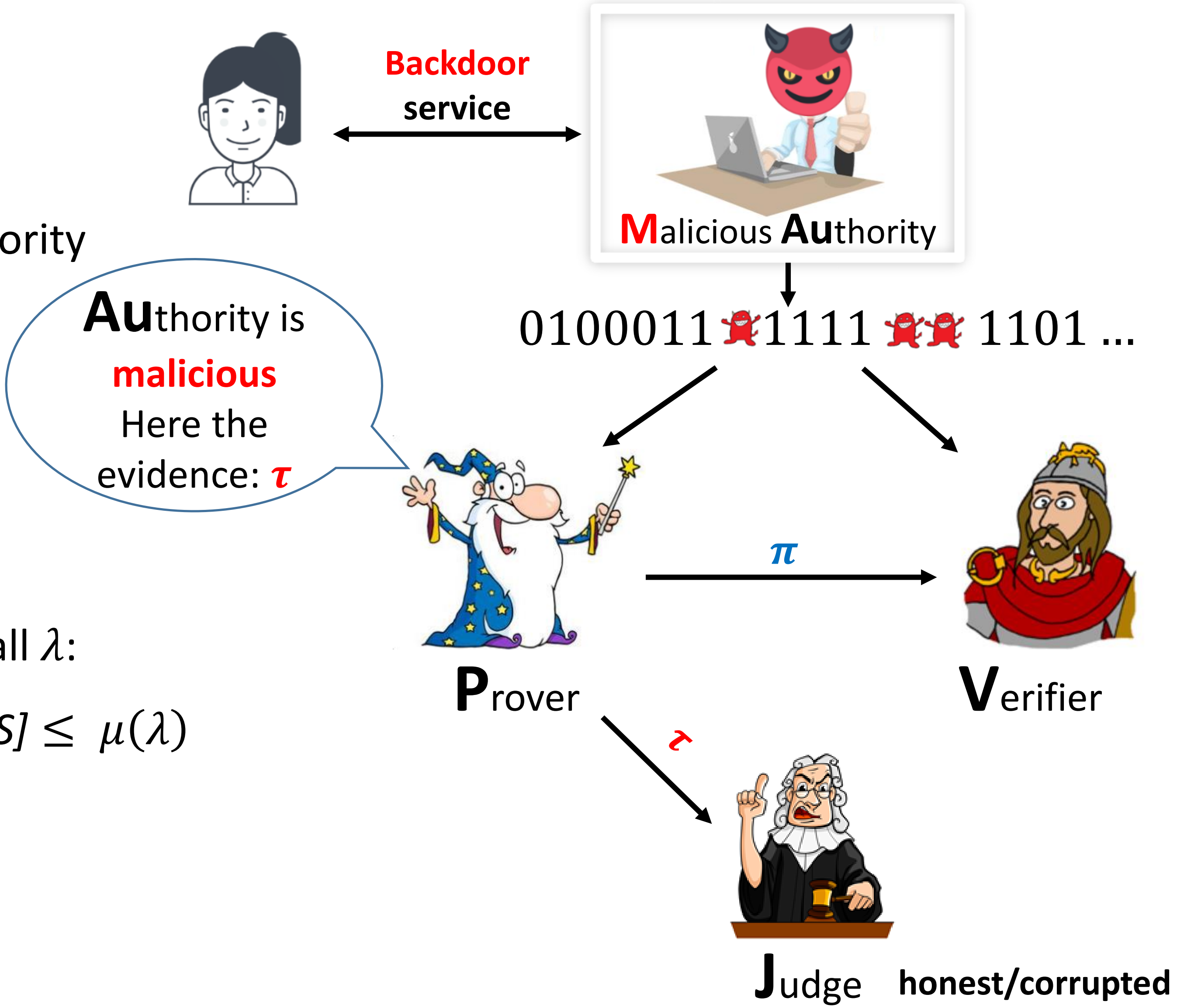
that is accepted by **Judge**.

Formally, $\forall$ PPT **malicious party** A, there

exists a negligible function $\mu(\cdot)$ such that for all λ:

$Pr[\textit{Judge}(CRS, A(CRS))\ \textit{outputs}\ \textbf{\textit{corrupted}}\ CRS] \leq \mu(\lambda)$

where $CRS \leftarrow GenCRS(1^{\lambda})$

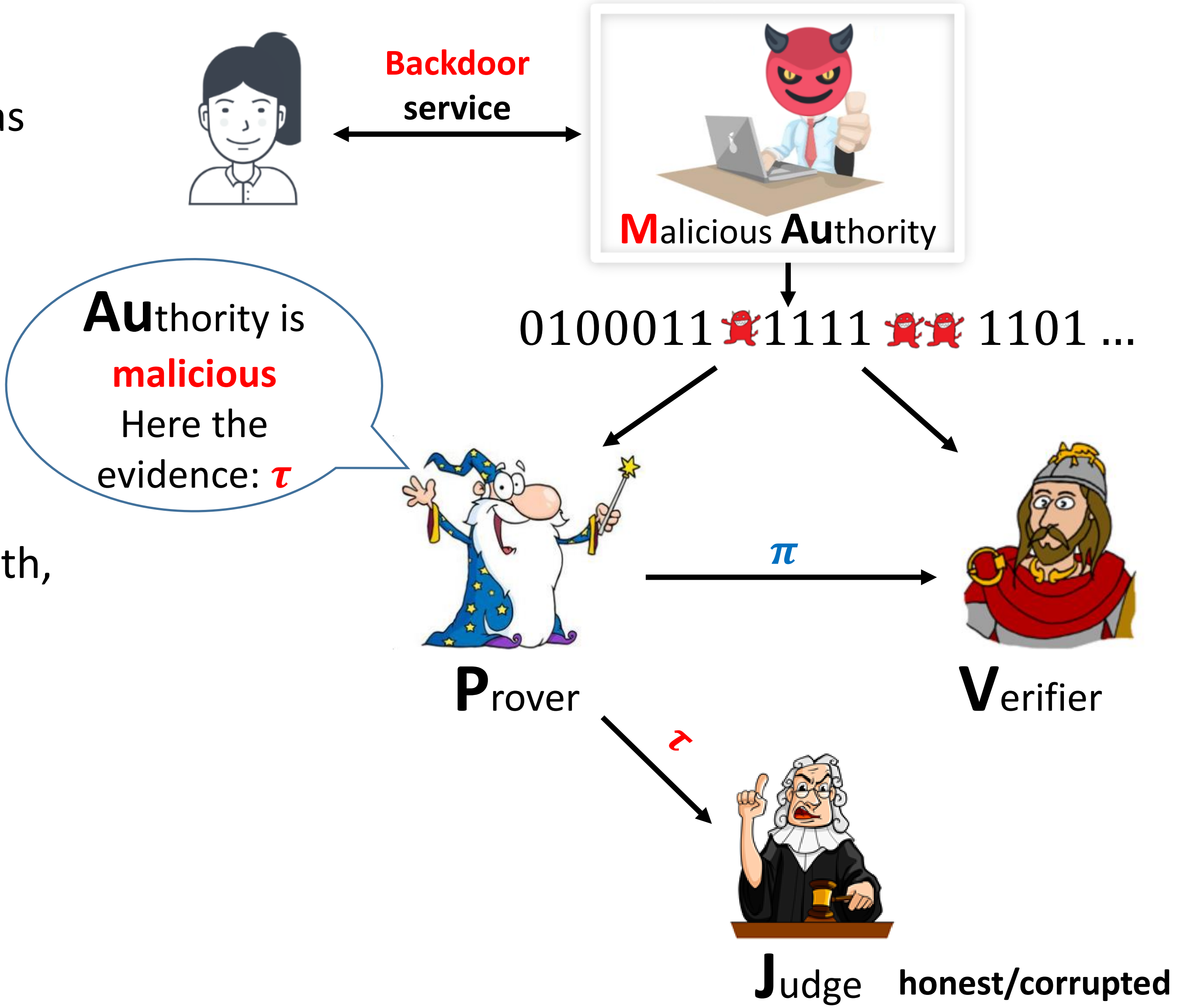


CRS generation in the real world

We say that **(GenCRS, Prove, Verify, Judge)** has

Malicious Authority Security for NIZK if:

- **(GenCRS, Prove, Verify)** is a NIZK proof system
- **(GenCRS, Prove, Verify, Judge)** satisfies both, **accountability** and **defamation free.**



Accountability

Acc.Real

Sample (x, w)

$\pi \leftarrow Prove(CRS^*, x, w)$

CRS^*

π

w'

Malicious

Authority

The **output** is 1 iff: $R(x, w') = 1$

Accountability

Acc.Real

Sample (x, w)

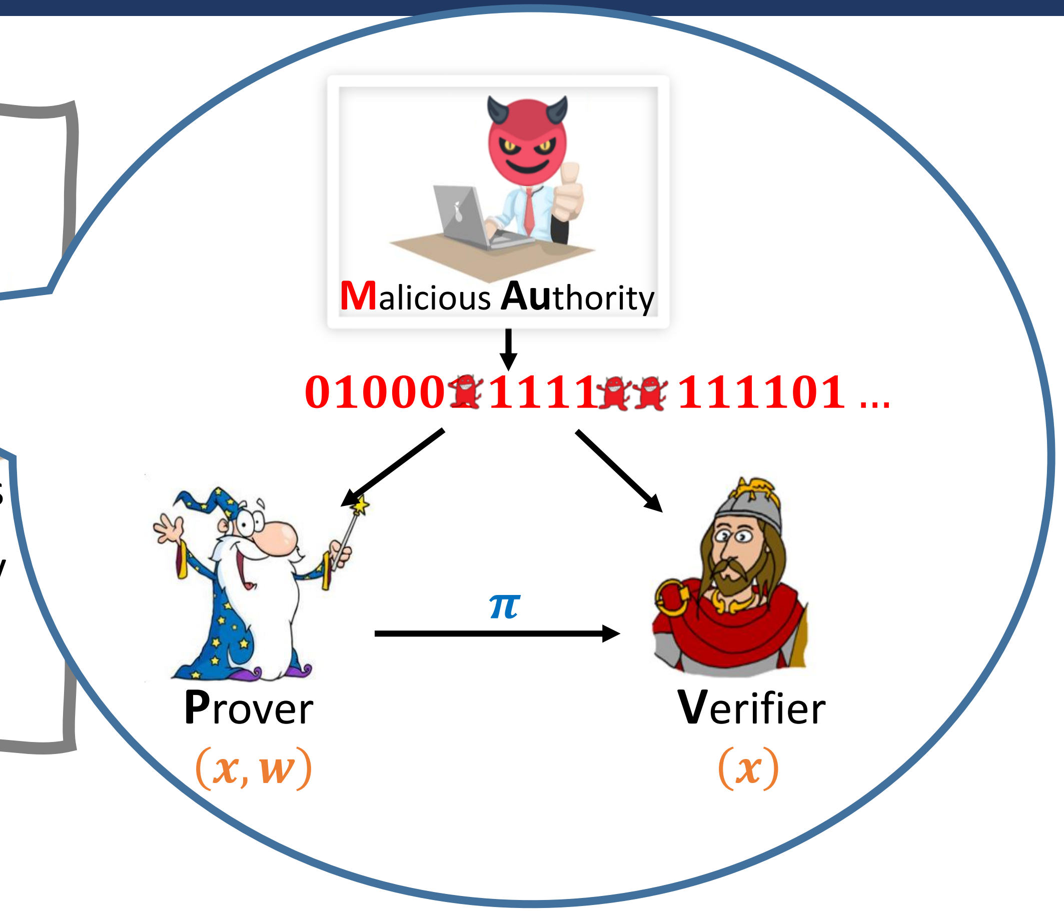
$\pi \leftarrow Prove(CRS^*, x, w)$

$\xleftarrow{CRS^*}$

$\xrightarrow{\pi}$

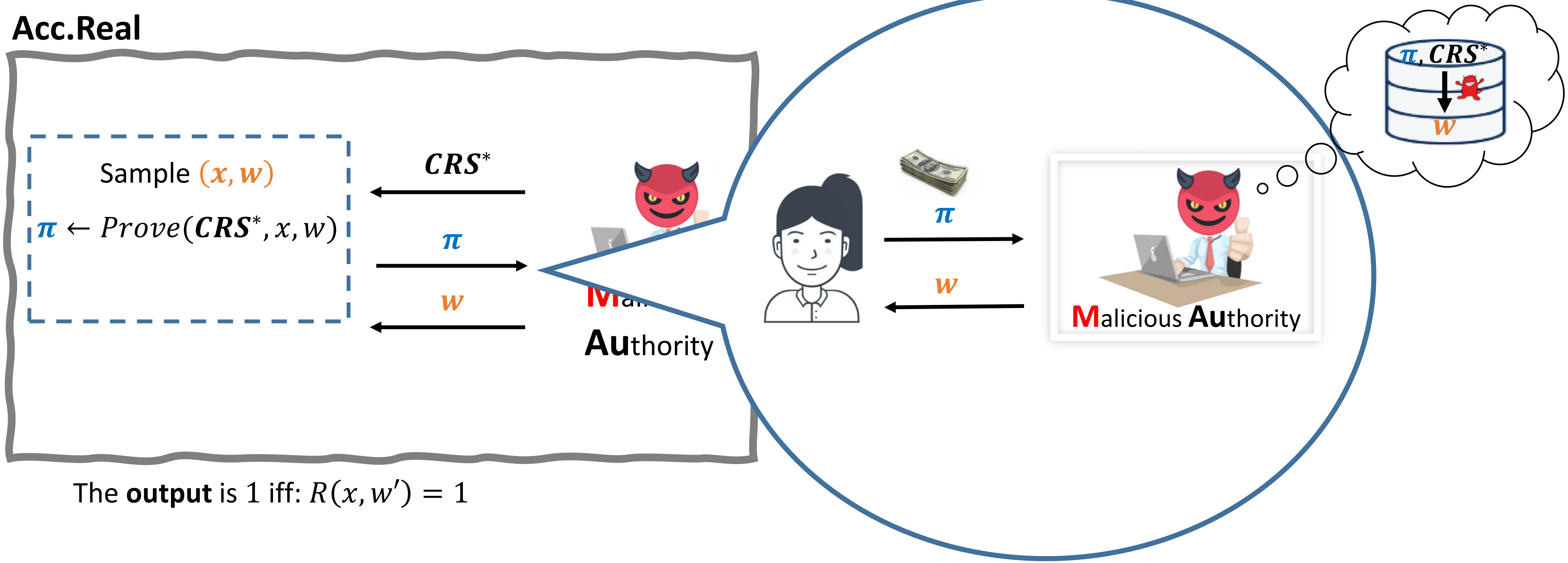
$\xleftarrow{w'}$

Malicious **Au**thority



The **output** is 1 iff: $R(x, w') = 1$

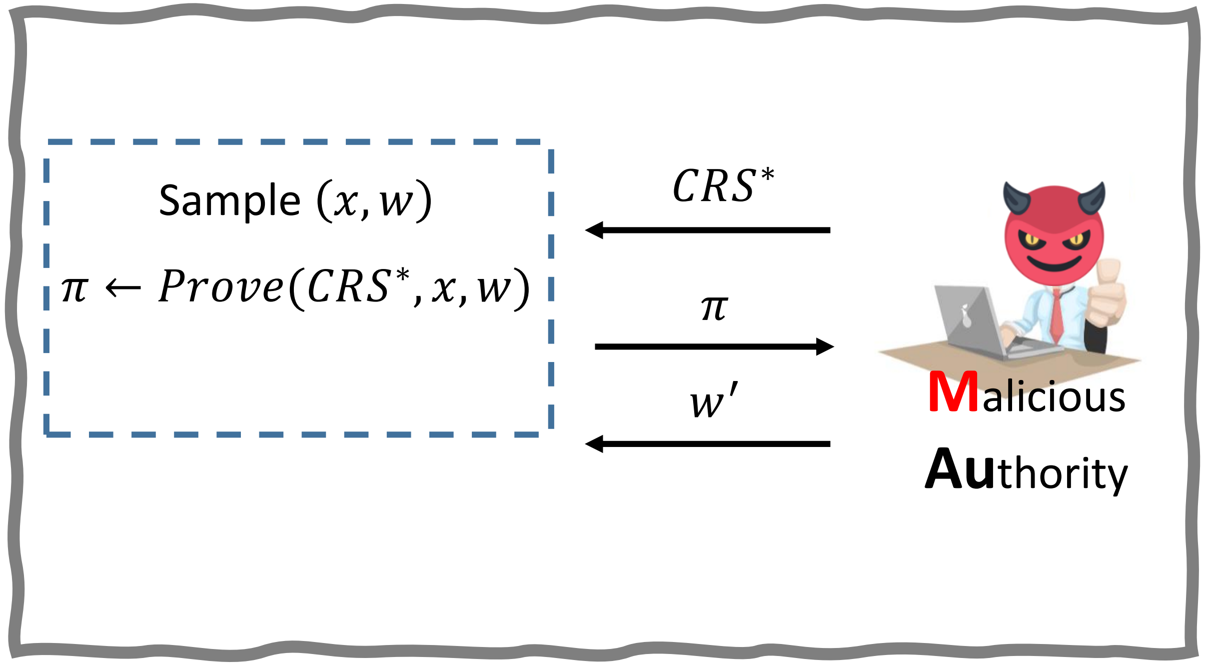
Accountability



The **output** is 1 iff: $R(x, w') = 1$

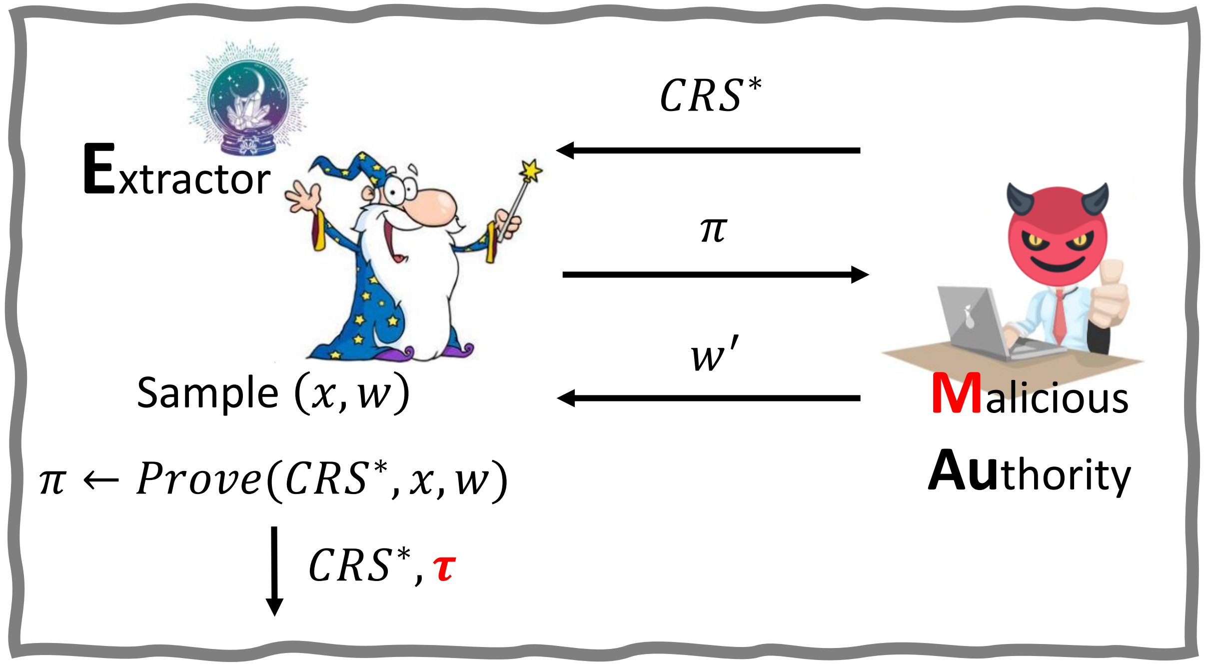
Accountability

Acc.Real

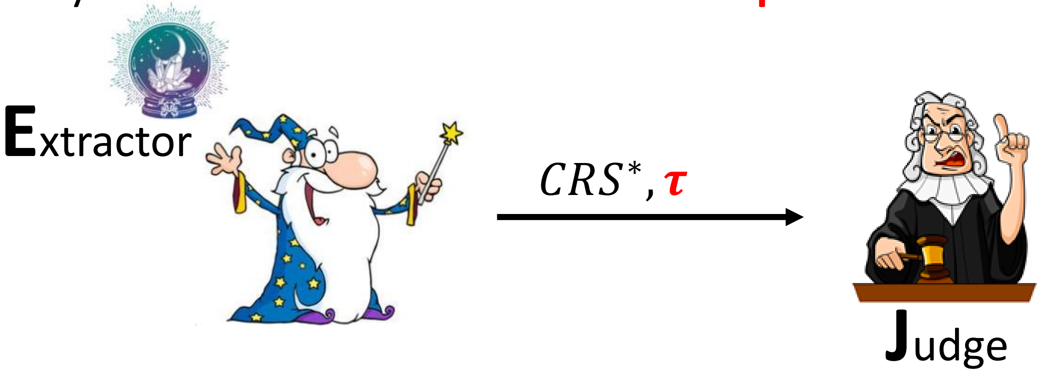


The **output** is 1 iff: $R(x, w') = 1$

Acc.Ext



The **output** is 1 if the **Judge** will be convinced by the **evidence $\boldsymbol{\tau}$** that CRS^* is **corrupted**



Accountability

Acc.Real

Sample (x, w)

$\pi \leftarrow Prove(CRS^*, x, w)$

CRS^*

π

w'

Malicious

Authority

The **output** is 1 iff: $R(x, w') = 1$

Acc.Ext

Extractor

CRS^*

π

w'

Malicious

Authority

Sample (x, w)

$\pi \leftarrow Prove(CRS^*, x, w)$

$CRS^*, \boldsymbol{\tau}$

The **output** is 1 if the **Judge** will be convinced by the **evidence** $\boldsymbol{\tau}$ that CRS^* is **corrupted**

Accountability: $\forall$ PPT **authority** A that succeeds in $\mathbf{Acc.Real}$, there exists an PPT **extractor** E that succeeds in $\mathbf{Acc.Ext}$

Our Results

Positive Results

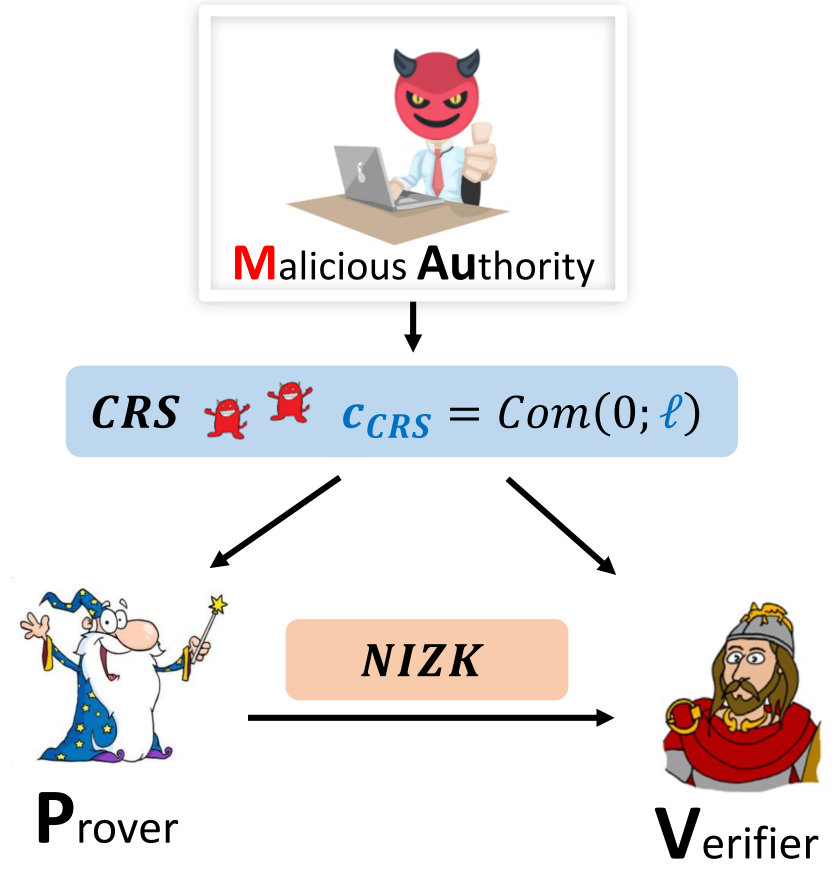
Theorem (Informal). Assuming SXDH on bilinear maps, there exists a NIZK for NP language in the CRS model satisfying both the **accountability** and the **defamation-free** properties.

High Level of Our Construction

Malicious Authority Security for NIZK

Starting point: Force the **CRS** **authority** to add a **commitment** to the CRS. Then, the **proof** is the ability to **open** the commitment.

If the authority is **malicious**, then from the **obtained witness** the **extractor** can **recover** the **secret** ℓ in the **CRS** and prove to the judge



Tools: Re-rendomizable bit commitment scheme [GOS06,ADKL19]

$Com(0; \ell)$ $\xrightarrow[\text{sample } r]{\textbf{Rerandomize}}$ $Com(0; \ell \oplus r)$

Malicious Authority Security for NIZK

Malicious Authority

CRS $c_{CRS} = Com(0; \ell)$

$NIZK\ of\ \hat{c}$

$\ell \oplus r$

Extractor

Prover

$NIZK\ of\ c$

Verifier

Statement: $c = Com(0; x)$

Witness: x

Toy example, not an NPC language

Sample r and **rerandomize**

$Com(0; \ell) \rightarrow Com(0; \ell \oplus r)$

Statement: $\hat{c} = Com(0; \ell \oplus r)$

Witness: $\ell \oplus r$

Malicious Authority Security for NIZK

Malicious Authority

CRS $c_{CRS} = Com(0; \ell)$

$NIZK$ of $\hat{c}$

$\ell \oplus r$

Extractor

ℓ, c_{CRS}

Judge

Extract ℓ

Statement: $\hat{c} = Com(0; \ell \oplus r)$

Witness: $\ell \oplus r$

Check: if $c_{CRS} = Com(0; \ell)$

Output: corrupted CRS

Prover

$NIZK$ of c

Verifier

Statement: $c = Com(0; x)$

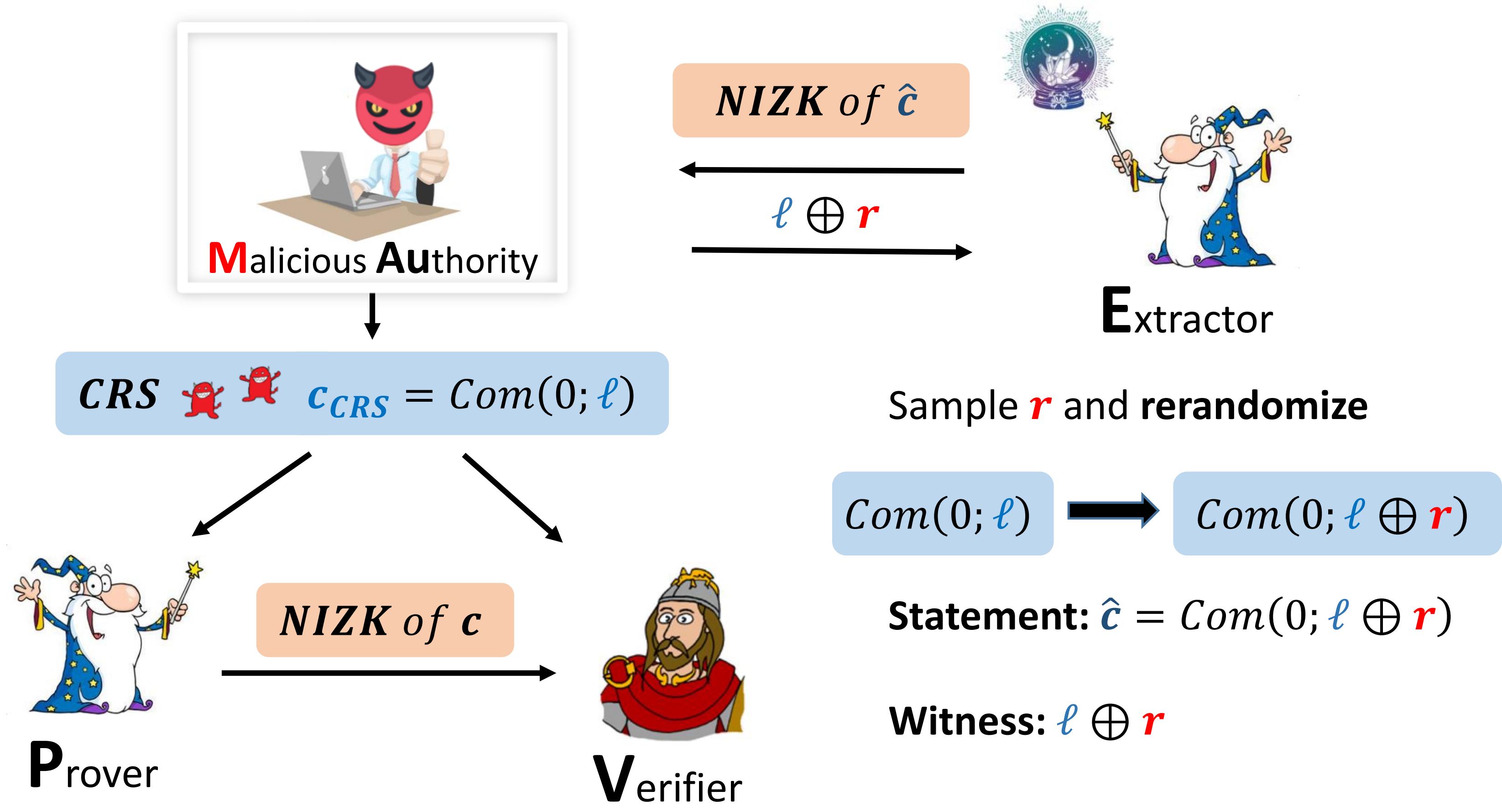
Witness: x

Toy example, not an NPC language

Malicious Authority Security for NIZK

Accountability follows from perfect rerandomization.

Defamation free follows from the security of the commitment.



Statement: $c = Com(0; x)$

Witness: x

Challenges

- In the paper, we extend this idea to an NPC problem (a variant of Circuit Satisfiability)
- A major challenge is to **generate** a NIZK while the **extractor does not** know the **witness**

Malicious **Au**thority

$\boldsymbol{CRS}$ $\quad c_{CRS} = Com(0; \ell)$

$\boldsymbol{NIZK\ of\ \hat{c}}$

$\longleftarrow$

$\ell \oplus \boldsymbol{r}$

$\longrightarrow$

Extractor

Sample $\boldsymbol{r}$ and **rerandomize**

$Com(0; \ell) \Longrightarrow Com(0; \ell \oplus \boldsymbol{r})$

Statement: $\hat{c} = Com(0; \ell \oplus \boldsymbol{r})$

Witness: $\ell \oplus \boldsymbol{r}$

Challenges

- Our approach is to force the **authority** to add more information to the CRS.

  However, if the **authority** is a **malicious** party, how can the prover **check** that the **additional information** is valid?

  - We cannot use NIZK since it will require CRS

More Results – Accountability in 2PC

2PC in CRS model

- We **cannot** achieve malicious 2 rounds 2PC in the plain model **[MW16, GS18, BL18]**
- In the CRS model, we **can** achieve malicious 2 rounds 2PC, but a corrupted authority can **recover** the **private inputs**

Can we achieve ***accountability*** *in CRS generation for* ***2PC****?*

- We extend the definition of accountability for 2PC

Strong Accountability

In 2PC protocol the **authority** can be **active** – and corrupted one of the parties during the protocol.

We call such a case **strong accountability**, and we ask whether **strong accountability** is achievable.

Our Results - OT

Positive Results

Theorem (Informal). Assuming IO for P/poly [BGI+01,GGH+16] and SXDH on bilinear groups, there exists a two-round maliciously secure OT in the CRS model satisfying both **strong accountability** and **defamation-free** properties.

Theorem (Informal). Assuming SXDH on bilinear maps, there exists a two-round maliciously secure OT in the CRS model satisfying both **weak accountability** and **defamation-free**.

Our Results – 2PC

Impossibility Result

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

Positive Results

Theorem (Informal). Assuming SXDH on bilinear maps, there exists a two-round maliciously secure two-party computation protocol for G satisfying both **weak accountability** and **defamation-free**.

* The class of functions G includes for instance: oblivious transfer, private information retrieval, subset sum, and more.

Our Results – 2PC

Impossibility Result

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

Positive Results

Theorem (Informal). Assuming SXDH on bilinear maps, there exists a two-round maliciously secure two-party computation protocol for G satisfying both **weak accountability** and **defamation-free**.