Implementing Resettable UC-functionalities with Untrusted Tamperproof Hardware-Tokens

Nico Döttling, Thilo Mie, Jörn Müller-Quade & Tobias Nilges

Karlsruhe Institute of Technology

Motivation

- [Katz07] introduced tamper-proof hardware as a setup-assumption
- Allows for UC-secure protocols which are not possible in the plain model
- Interaction can be shifted from one party to the token, making protocols non-interactive*
- Stateful token: OT is possible

*for the sender

Untrusted resettable hardware

- What happens if the token is resettable?
- We know we can make most protocols resettably secure with standard techniques (e.g. [CGGM00])
- Use some general purpose MPC-compiler (e.g. CLOS02) to get UC-security for MPC

➡ We get non-interactive resettable UC-secure MPC

CRS suffices for this!

Our Results

- Open Question: How to implement a CRS with untrusted resettable tamper-proof hardware?
- Our Results:
  - CRS with a single resettable token and an interactive initialization phase
  - Non-interactive protocol for a *resettable* CRS with two resettable tokens
  - Impossibility result for non-interactive protocols with a single resettable token

Related Work

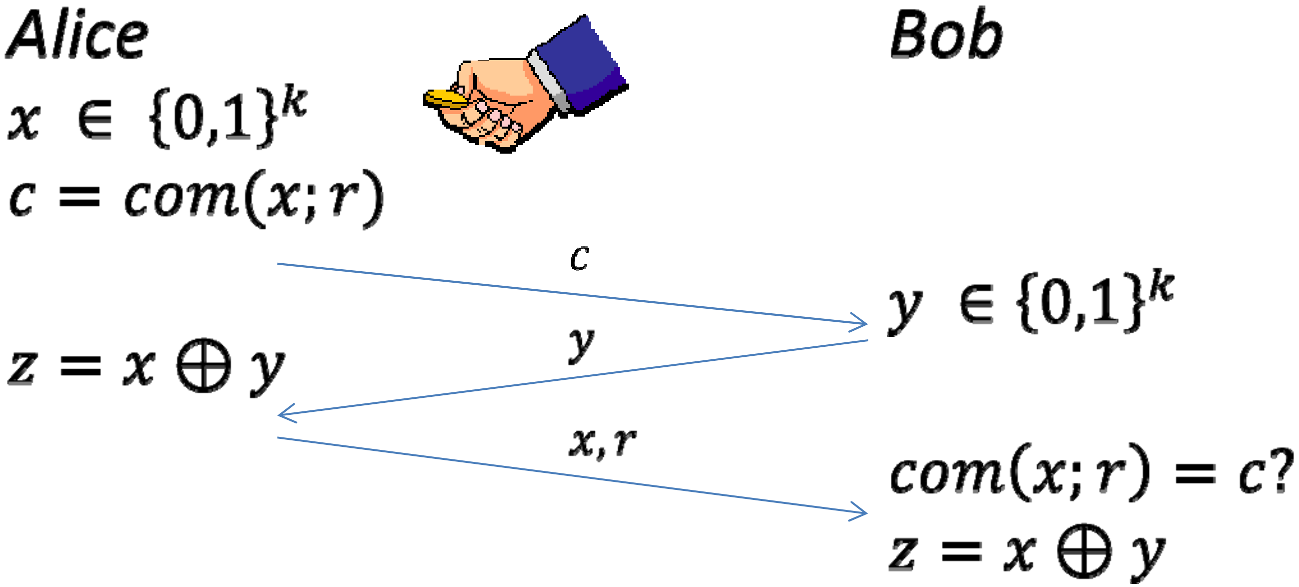
UC-secure 2PC using stateless hardware

- [CGS08]
  - Assuming OT in the plain model
  - Requires interaction
- [GIS+10]
  - Several tokens for interactive case
  - Non-interactive protocol with semi-honest sender

- CRS protocol similar to ours independently presented by [CKS+11], but not the non-interactive case

Starting Point

- Blum coin toss



Alice

$x \in \{0,1\}^k$

$c = com(x; r)$

c

Bob

$y \in \{0,1\}^k$

y

$z = x \oplus y$

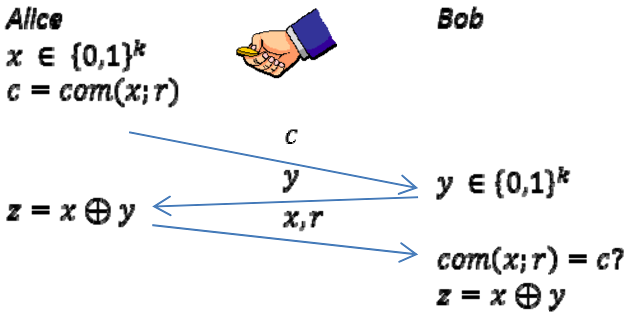
x, r

$com(x; r) = c?$

$z = x \oplus y$

CRS with one resettable token

- Basic idea: Blum coin toss using the resettable token as the commitment
- Problem: Token must reveal the coins of Alice only after Bob sent his coins to Alice
- Solution: Lock the token with a password



Alice

$x \in \{0,1\}^k$

$c = com(x;r)$

Bob

c

$y \in \{0,1\}^k$

y

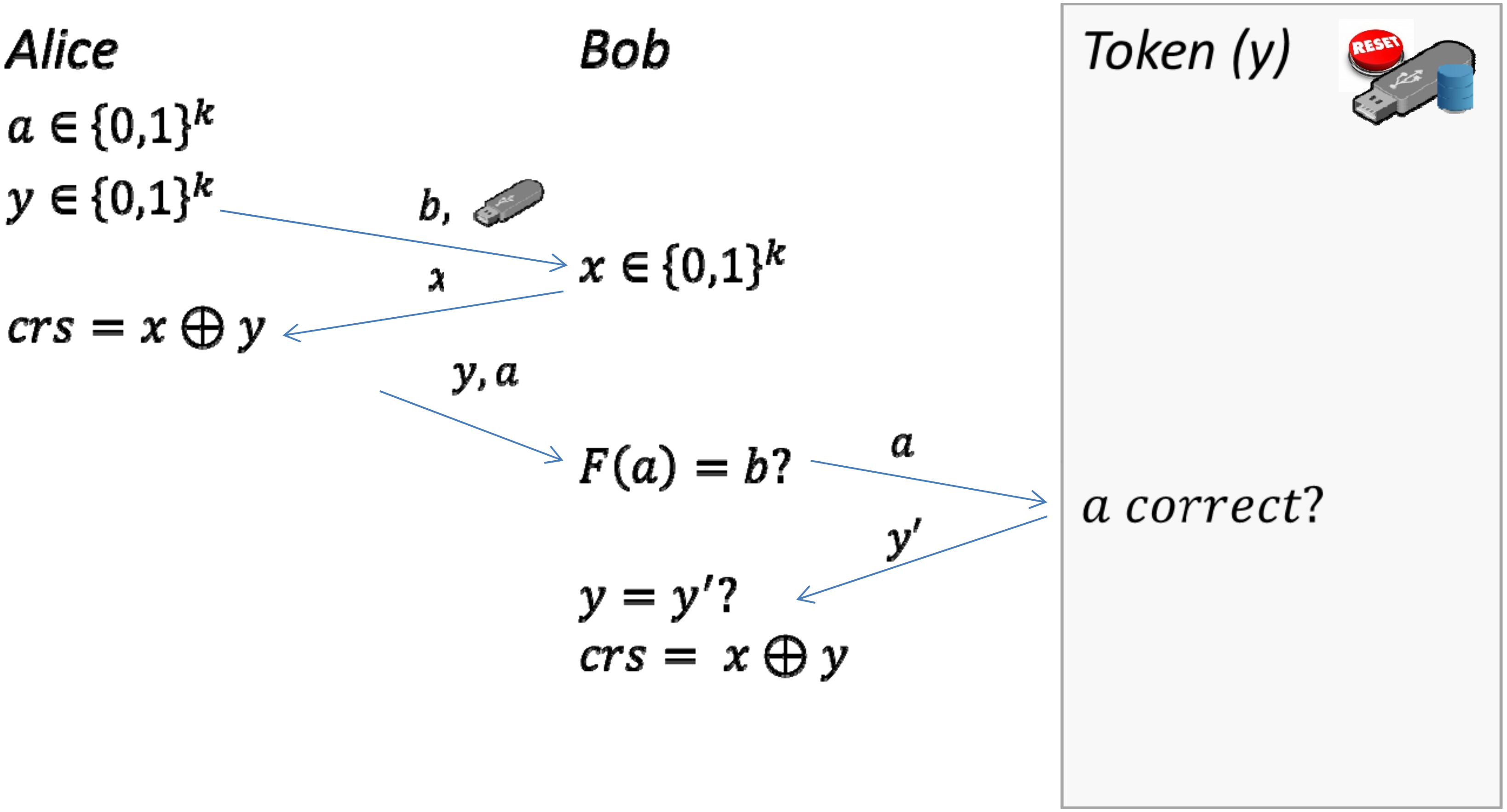
$z = x \oplus y$

x,r

$com(x;r) = c?$

$z = x \oplus y$

First Try



Alice

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$b,$

Bob

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

y, a

$F(a) = b?$

a

Token (y)

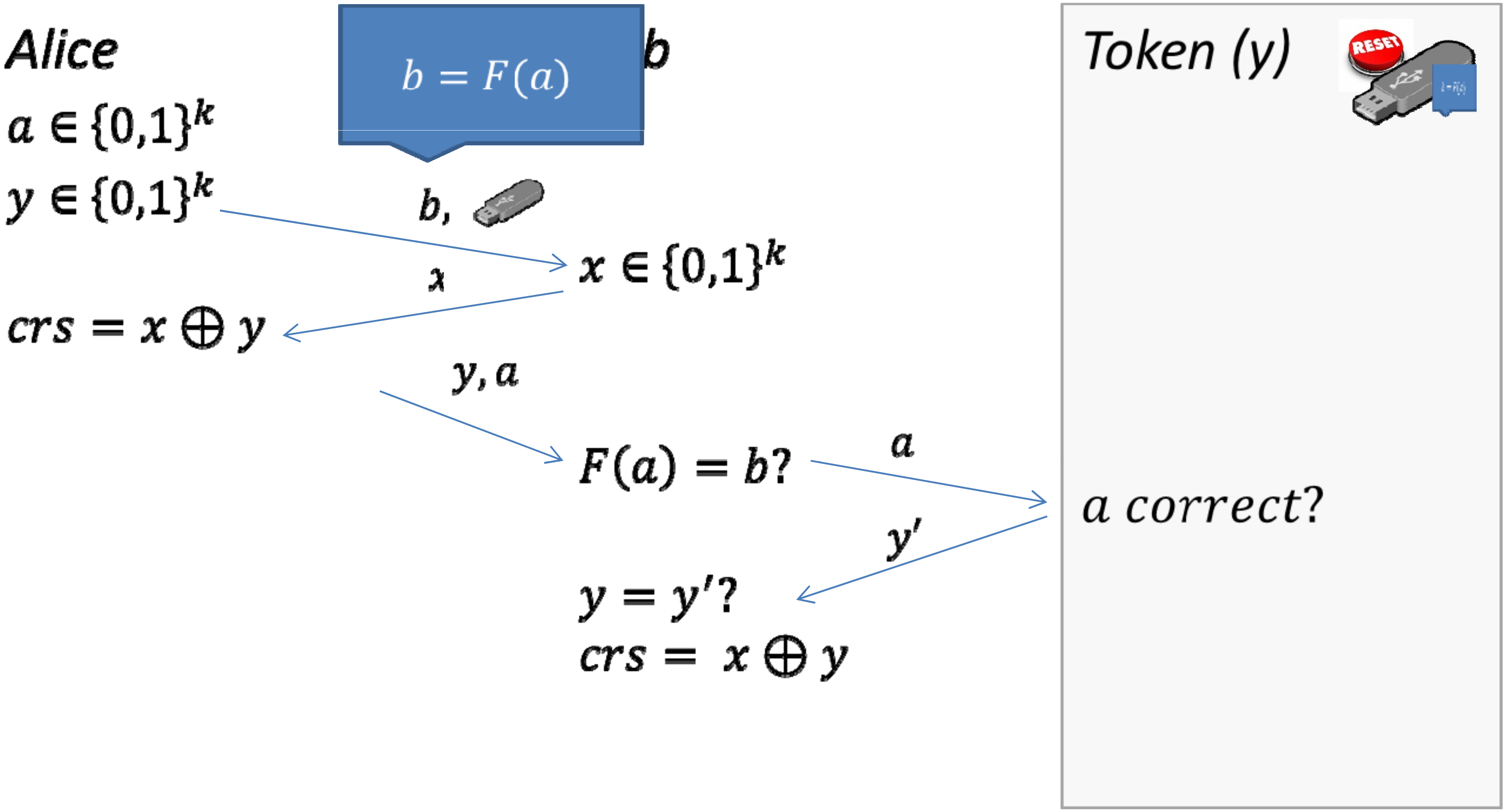
$a\ correct?$

y'

$y = y'?$

$crs = x \oplus y$

First Try



Alice

$b = F(a)$

b

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$b,$

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

y, a

$F(a) = b?$

a

Token (y)

RESET

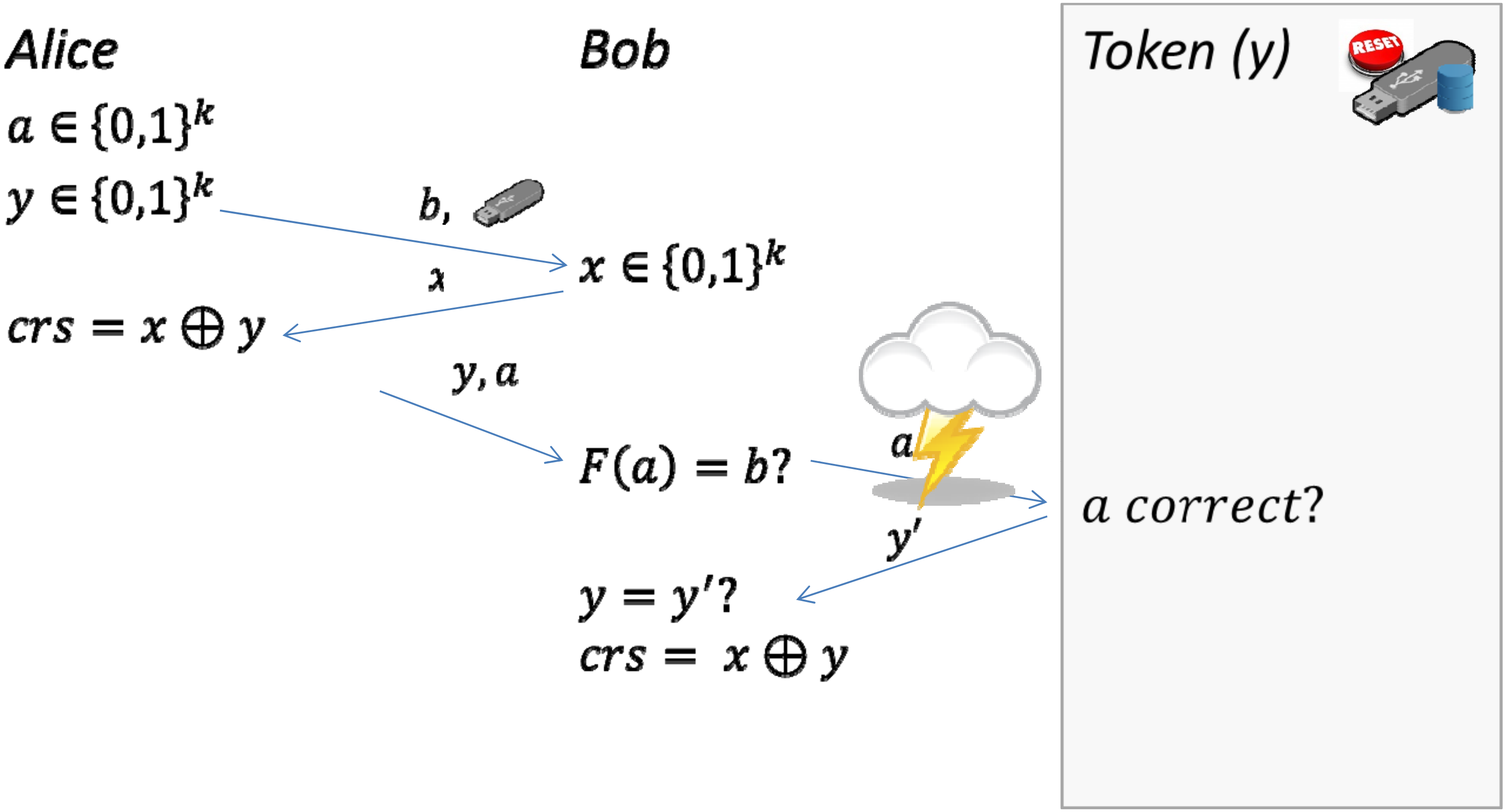
$a\ correct?$

y'

$y = y'?$

$crs = x \oplus y$

First Try



Alice

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$b,$

Bob

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

y, a

$F(a) = b?$

a

Token (y)

a correct?

y'

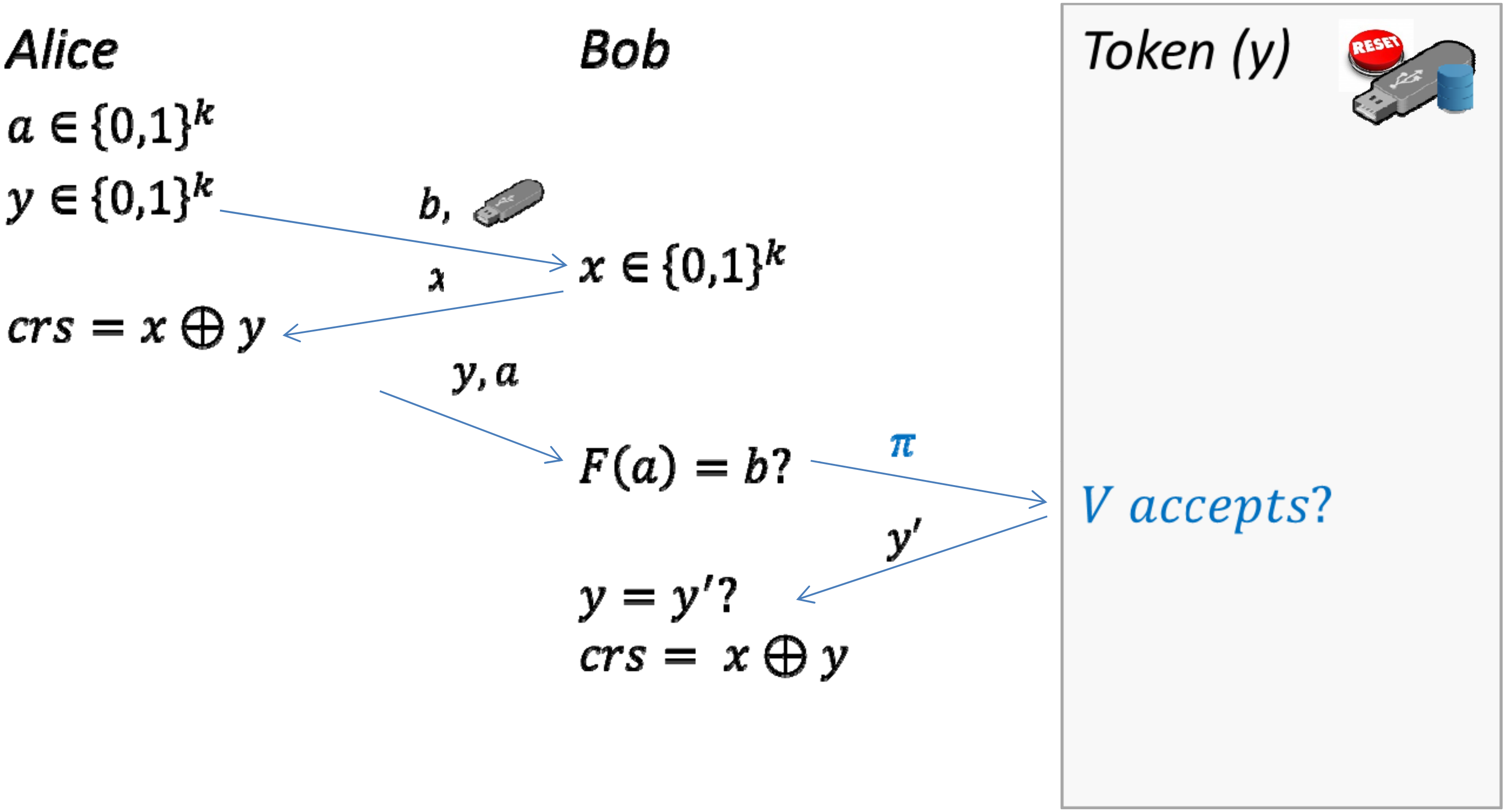
$y = y'?$

$crs = x \oplus y$

CRS with one resettable token

- Problem:
  - Not simulatable
  - We want to extract the secret from the token without knowing the password *a*
- Solution: Use a resettably-sound zero knowledge argument of knowledge

Second Try



Alice

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$b,$

Bob

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

y, a

$F(a) = b?$

π

y'

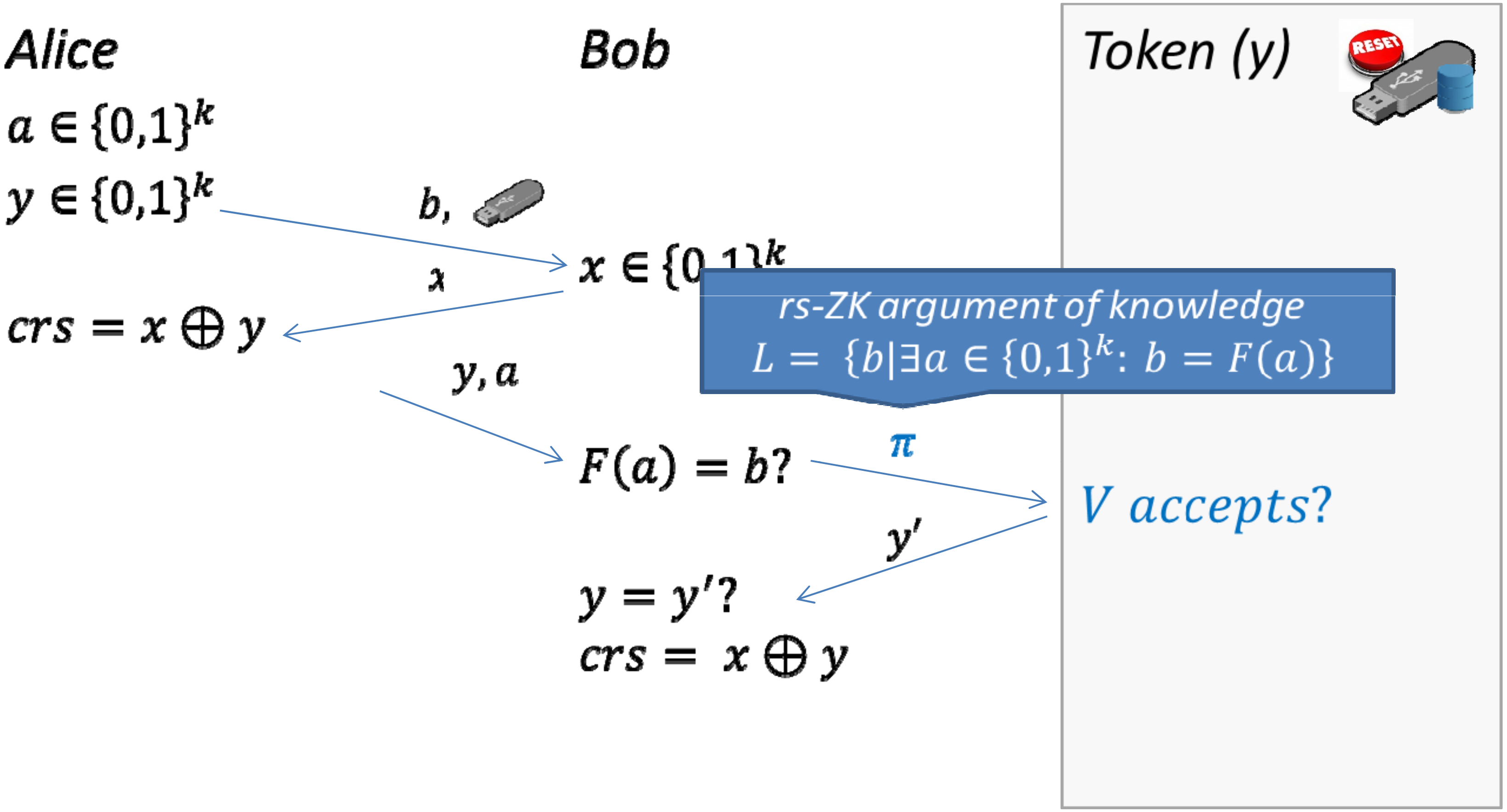
$y = y'?$

$crs = x \oplus y$

Token (y)

$V\ accepts?$

Second Try



Alice

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$b,$

Bob

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

y, a

$F(a) = b?$

Token (y)

RESET

rs-ZK argument of knowledge

$L = \{b | \exists a \in \{0,1\}^k : b = F(a)\}$

π

V *accepts?*

y'

$y = y'?$

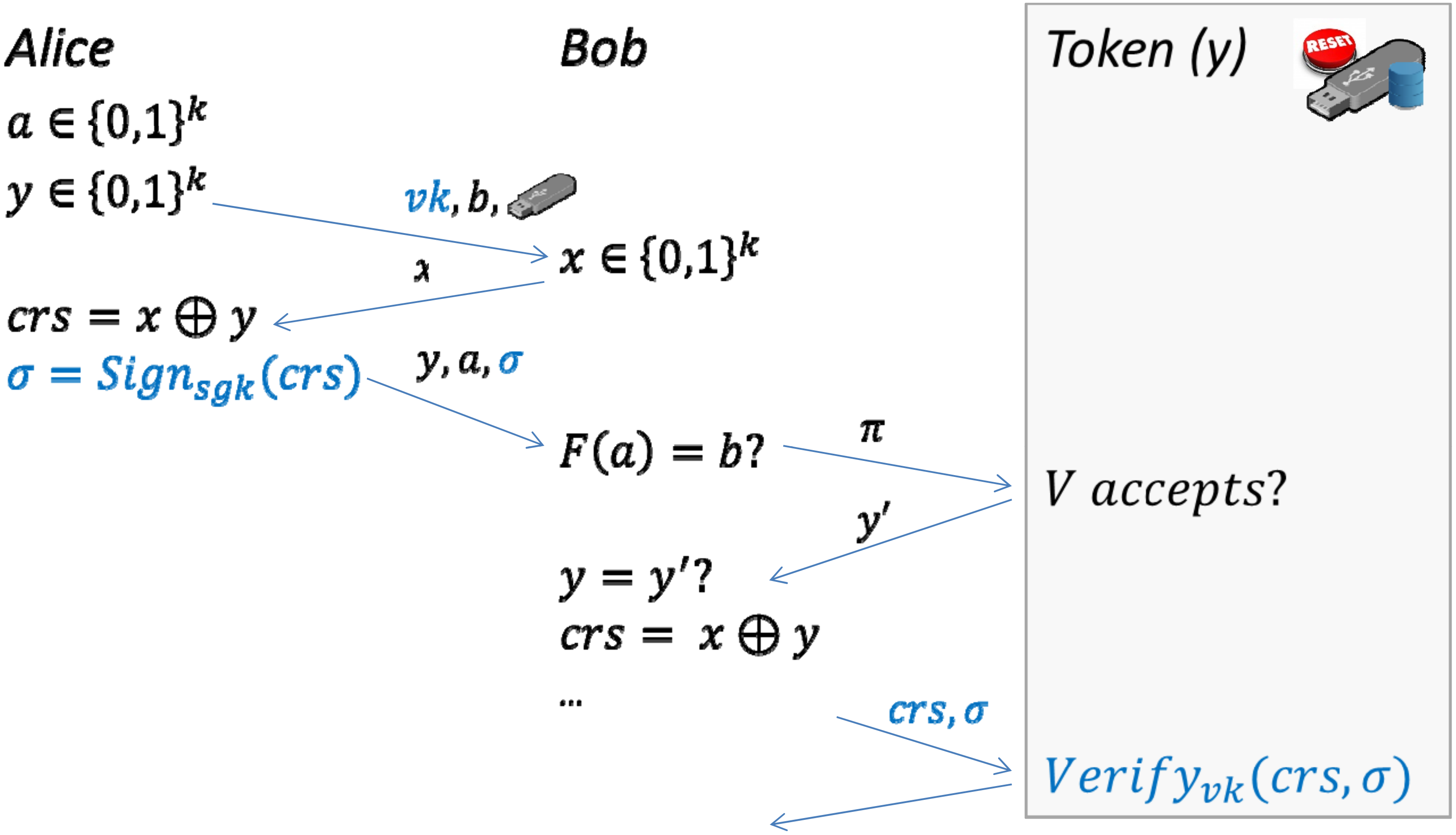
$crs = x \oplus y$

CRS with one resettable token

- What do we have:
  - we can implement a CRS with a resettable token
  - we only need a one-time initialization phase
  - it is UC-secure (we will come to this later)
  - But: Token has to be convinced that the CRS is valid

- Solution: We use a signature on the CRS and can just let the token verify the signature

Final Protocol



Alice

$a \in \{0,1\}^k$

$y \in \{0,1\}^k$

$vk, b,$

Bob

$x \in \{0,1\}^k$

x

$crs = x \oplus y$

$\sigma = Sign_{sgk}(crs)$

y, a, σ

$F(a) = b?$

π

Token (y)

$V\ accepts?$

y'

$y = y'?$

$crs = x \oplus y$

…

crs, σ

$Verify_{vk}(crs, \sigma)$

Proof Idea

- Goal: Simulator has to be able to arbitrarily choose the CRS

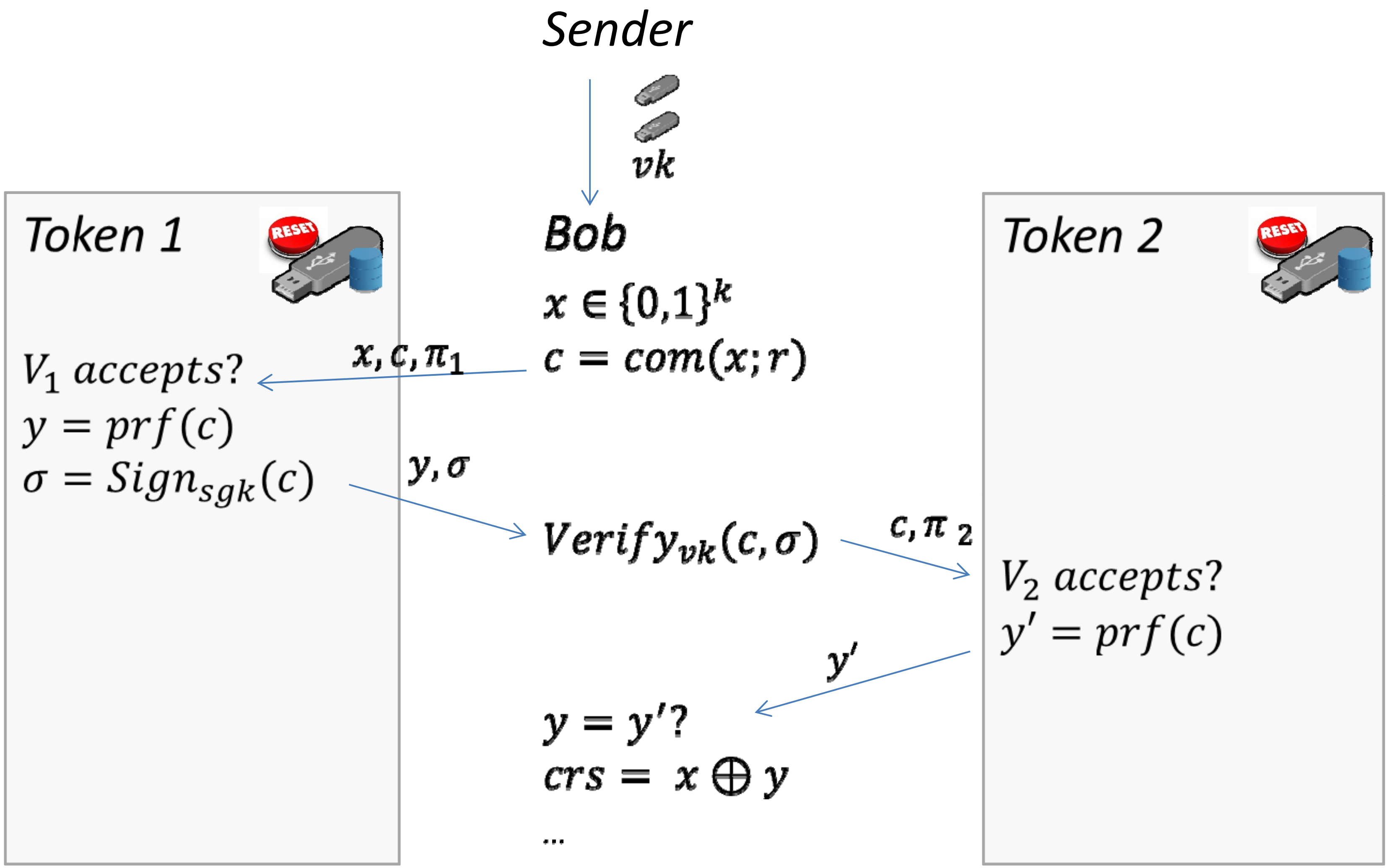
- Corrupted Receiver:
  - Simulator has joint view of sender and token
  - Simulator is not a priori commited to its coins
  - Sets $y = x \oplus crs$ after receiving Bobs coins

- Corrupted Sender:
  - Simulator simulates protocol out of order
  - Simulator first constructs a malicious verifier V^* for the rs-ZK AoK using the source code of the token
  - Uses the non-black-box simulator on $V^*$and b to obtain y
  - Then sets $x = y \oplus crs$ and proceeds normally

CRS with two resettable tokens

- We replace the sender with another resettable token
- Problem:
  - Previous approach fails here
  - Once the receiver learns *a*, it can learn *y* and then reset the token
  - CRS can be chosen by the adversary!
- Solution:
  - Replace the sender-coins with a pseudorandom function
  - The receiver has to commit to its input
  - The Token no longer sends a password but signs the commitment
  - Signature is used to unlock the second token instead of password

CRS with two resettable tokens



Sender

vk

Bob

$x \in \{0,1\}^k$

$c = com(x; r)$

x, c, π_1

Token 1

V_1 *accepts?*

$y = prf(c)$

$\sigma = Sign_{sgk}(c)$

y, σ

$Verify_{vk}(c, \sigma)$

c, π_2

Token 2

V_2 *accepts?*

$y' = prf(c)$

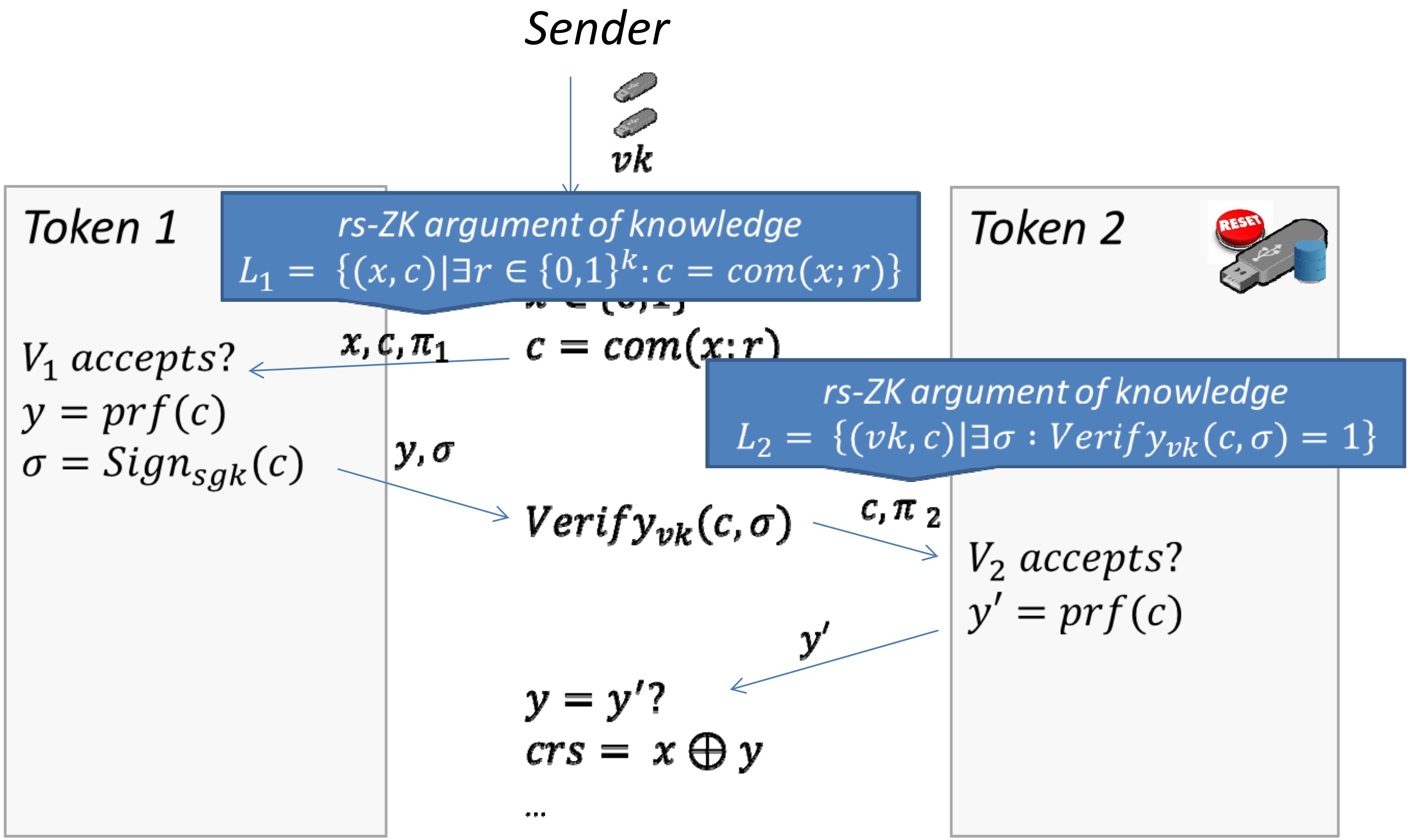
y'

$y = y'?$

$crs = x \oplus y$

…

CRS with two resettable tokens



Sender

vk

Token 1

Token 2

rs-ZK argument of knowledge

$L_1 = \{(x,c) \mid \exists r \in \{0,1\}^k : c = com(x;r)\}$

$x \in \{0,1\}$

$c = com(x;r)$

x, c, π_1

V_1 $accepts?$

$y = prf(c)$

$\sigma = Sign_{sgk}(c)$

y, σ

rs-ZK argument of knowledge

$L_2 = \{(vk,c) \mid \exists \sigma : Verify_{vk}(c,\sigma) = 1\}$

$Verify_{vk}(c,\sigma)$

$c, \pi\ _2$

V_2 $accepts?$

$y' = prf(c)$

y'

$y = y'?$

$crs = x \oplus y$

…

Impossibility Result

- Non-interactively implementing a point function with a single resettable token is not possible!
- A successful simulator for a corrupted token directly yields a cheating strategy in the real world
- Even if more than one token is used, non-black-box techniques have to be used (which is expexted)

Summary

- We presented two protocols for CRS-generation based on a Blum coin toss
  - with a single resettable token and an interactive initialization phase
  - non-interactively with two resettable tokens
  - Optimal w.r.t. communication complexity and # of tokens
- Non-interactively creating a CRS with a single resettable token is not possible

Thank You!