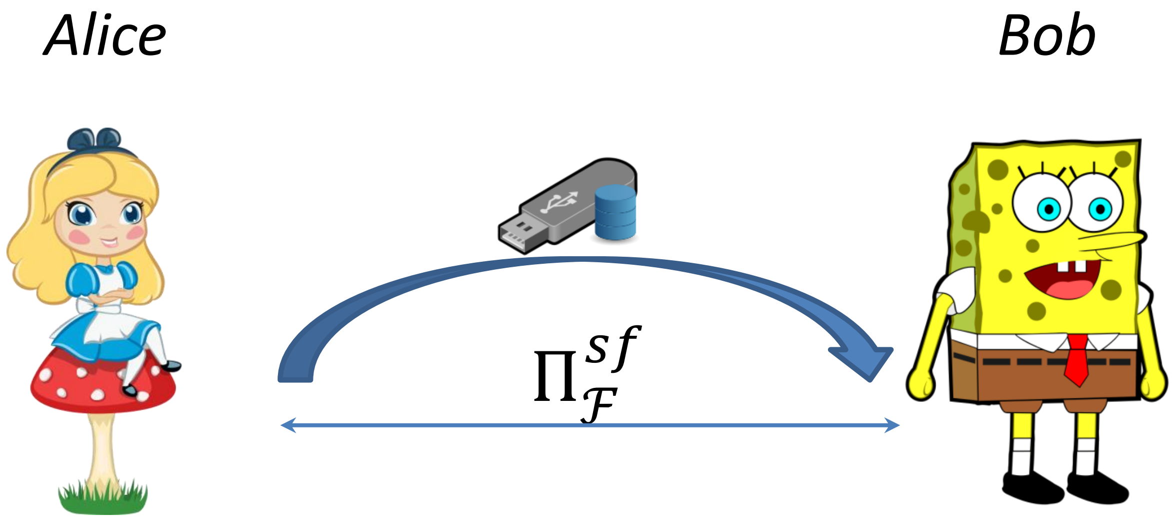
From Stateful to Resettable Hardware Using Symmetric Assumptions

Nico Döttling, Daniel Kraschewski, Jörn Müller-Quade & Tobias Nilges

Aarhus University, TNG Technology Consulting GmbH, Karlsruhe Institute of Technology

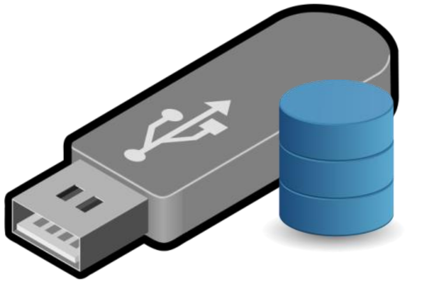
General Setting



$\mathcal{F}$ is arbitrary functionality, e.g. OT, Commitment….

Motivation

- UC-secure protocols impossible without setup-assumptions
- [Katz07] introduced tamper-proof hardware as a UC setup-assumption
- Stateful token: statistically UC-secure OT is possible [DKM11]



What about resettable tokens?

- Still powerful, but most statistically secure protocols impossible [GIMS10]
- Feasibility of NI-2PC for resettable functionalities shown by [DMMN13]
- Open question: relation between stateful and resettable token protocols wrt feasibility?

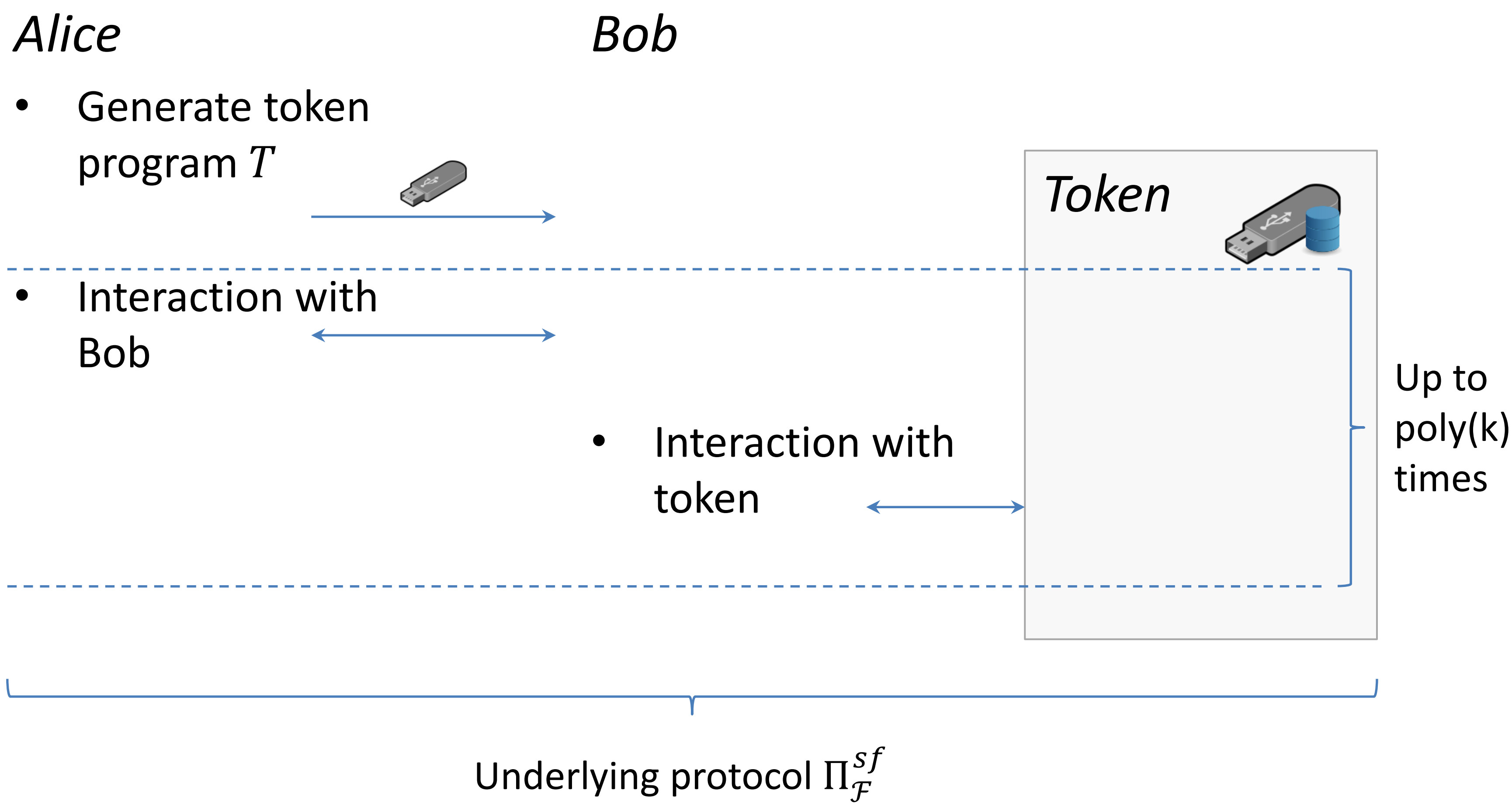
Our Results

- All protocols based on stateful tokens can be transformed to use resettable tokens
- General compiler for UC-secure protocols
  - Requires interaction
  - Requires computational assumptions (only OWF!) or additional setup

Basic Idea

- Shift state from token to Alice:
  - Alice authenticates inputs
  - Bob sends authenticated value to token
- Problem: Alice must not learn Bob‘s inputs
- Solution:
  - Alice authenticates encoding of input
  - Bob provides authentication and decoding information

Blueprint of Our Approach



Blueprint of Our Approach

Alice

Bob

- Generate token program T

Token

- Interaction with Bob

- Interaction with token

Up to poly(k) times

Blueprint of Our Approach

Alice

Bob

- Generate token program T
- Enhanced program T'

- Interaction with Bob

Token

- Interaction with token

Up to poly(k) times

Blueprint of Our Approach

Alice

Bob

- Generate toke program T
- Enhanced program T'

Answers only authen-ticated queries

- Interaction with Bob

Resettable token

Token

- Interaction with token

Up to poly(k) times

Blueprint of Our Approach

Alice

Bob

- Generate token program T
- Enhanced program T'

- Interaction with Bob

$enc(inp)$

- Authentication step

σ_{inp}

Token

- Interaction with token

Up to poly(k) times

Blueprint of Our Approach

Alice

- Generate token program T
- Enhanced program T'

- Interaction with Bob

Alice obliviously authenticates input

$enc(inp)$

σ_{inp}

- Authentication step

Bob

- Interaction with token

Token

Up to poly(k) times

Blueprint of Our Approach

Alice

- Generate token program T
- Enhanced program T'

Bob

- Interaction with Bob
- Authentication step

$enc(inp)$

σ_{inp}

- Verify authentication
- Interaction with token

Token

Up to poly(k) times

Blueprint of Our Approach

Alice

- Generate token program T
- Enhanced program T'

- Interaction with Bob
- Authentication step

$enc(inp)$

σ_{inp}

Bob

- Verify authentication
- Interaction with token

Token

Up to poly(k) times

New protocol $\Pi_{\mathcal{F}}^{res}$

Blueprint of Our Approach

Alice

- Generate token program T
- Enhanced program T'

Bob

- Interaction with Bob
- Authentication step

$enc(inp)$

σ_{inp}

Caution: channel from Alice to token

- Verify authentication
- Interaction with token

Token

Up to poly(k) times

New protocol $\Pi_{\mathcal{F}}^{res}$

Two Solutions

- Using resettably-sound zero-knowledge
  - Non-black-box, but necessary [DMMN13, CKS+14]
- OT-hybrid model
  - Allows only a fixed number of messages
  - Inf.-th. transformation

Solution Based on resettably-sound ZK

Alice

Bob

- Generate T according to $\Pi_{\mathcal{F}}^{sf}$
- Add ´functionality`

Token(T)

$c \leftarrow Commit(inp; r)$

c

$\sigma \leftarrow \mathrm{Sign}_{sgk}(c)$

σ

$\pi \leftarrow P(c, \sigma, r, inp)$

π, inp

$r' \leftarrow PRF(inp)$

$V(\pi; r') = 1?$

out

$out \leftarrow T(inp)$

Up to poly(k) times

Proof Idea

- Every adversary against$\Pi_{\mathcal{F}}^{res}$can be transformed into adversary against $\Pi_{\mathcal{F}}^{sf}$
- $\Pi_{\mathcal{F}}^{sf}$ is UC-secure by assumption

- Corrupted Receiver:
  - Simulator has joint view of sender and token
  - Locally performs all checks that the token would perform
  - If checks are OK, proceed like in $\Pi_{\mathcal{F}}^{sf}$

- Corrupted Sender:
  - Simulator has to input token code of $\Pi_{\mathcal{F}}^{res}$ into stateful token
  - Simulator first constructs $\hat{T}$:
    - Use source code of T^{res} to create V^*
    - Use non-black-box simulator on V^* to generate fake proof
    - Upon input, fake proof and proceed with execution of T^{res}

Efficiency

- ZK proof for each token input, but
  - Typically constant round protocols…
  - Some protocols allow non-adaptive inputs
- Non-adaptive inputs: create hash-tree of queries and authenticate root
  - CRHF > OWF!
  - Use Sig-Com Trees [CPS13], based on OWF

Implications

- Apply compiler to [DKM11] to obtain most efficient UC-secure OT-protocol from OWF
  - Token sent in one direction only
  - Constant round
  - Very efficient ([DKM11] provides inf-th. security)
- In OT-hybrid model, even inf-th. protocols can be realized

Thank You!