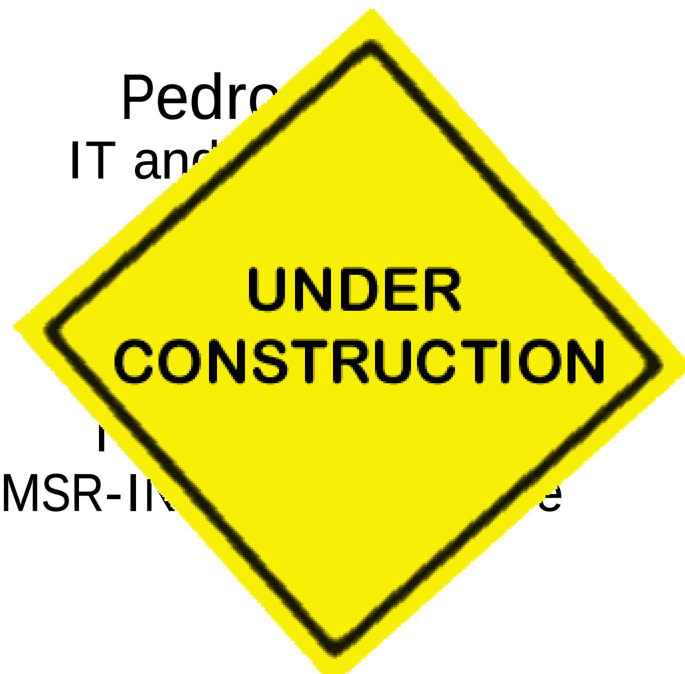
CoSyProofs, April6-9. 2009

High-Level Programming for E-Cash

Pedr
IT an

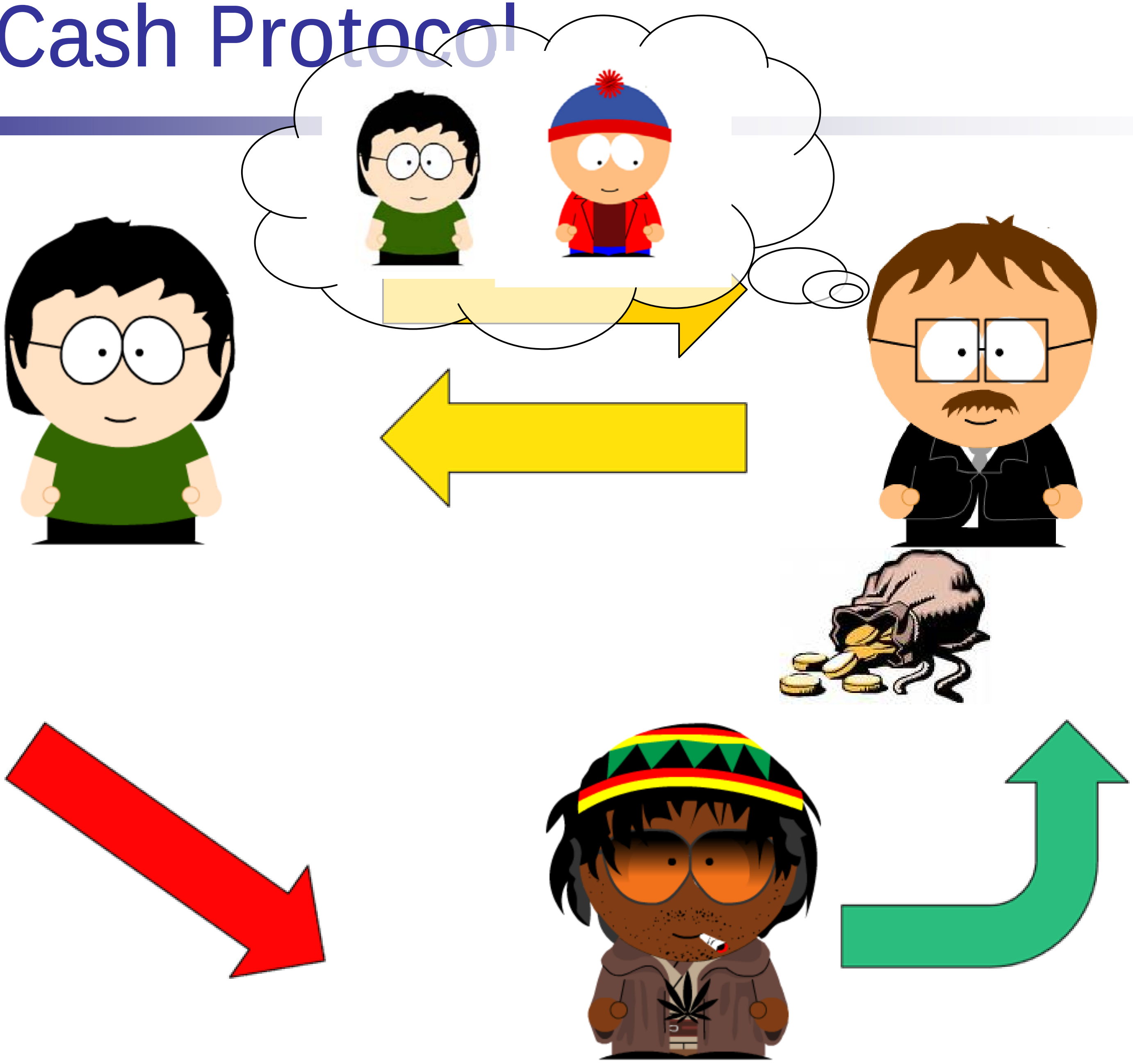


elli
MSR-I e

E-Cash Protocols

- Introduced by Chaum in 1982 it intends to simulate the use of traditional money
  - Many interesting properties of the traditional money are delicate to mimic in digital world
- **Aim** to provide robust abstractions for anonymous payment protocols
  - Users should spend coins anonymously
  - Users cannot forge coins
  - User should not spend the same coin twice without being eventually caught
  - Spending should be offline
- By necessity, these protocols involve sophisticated cryptographic constructions

E-Cash Protocol



E-Cash Protocol Spec

- Withdraw($\mathcal{U}(pk_\mathcal{B}, sk_\mathcal{U}, n), \mathcal{B}(pk_\mathcal{U}, sk_\mathcal{B}, n)$) protocol, the user $\mathcal{U}$ withdraws a wallet W of n coins from the bank $\mathcal{B}$. The users output is the wallet W, or an error message. $\mathcal{B}$s output is some information T_W which will allow the bank to trace the user should this user double-spend some coin, or an error message. The bank maintains a database D for this trace information, to which it enters the record $(pk_\mathcal{U}, T_W)$.

- Spend($\mathcal{U}(W, pk_\mathcal{M}), \mathcal{M}(sk_\mathcal{M}, pk_\mathcal{B}, n)$) protocol, a user $\mathcal{U}$ gives one of the coins from his wallet W to the merchant $\mathcal{M}$. Here, the merchant obtains a serial number S of the coin, and a proof π of validity of the coin. The users output is an updated wallet W'.

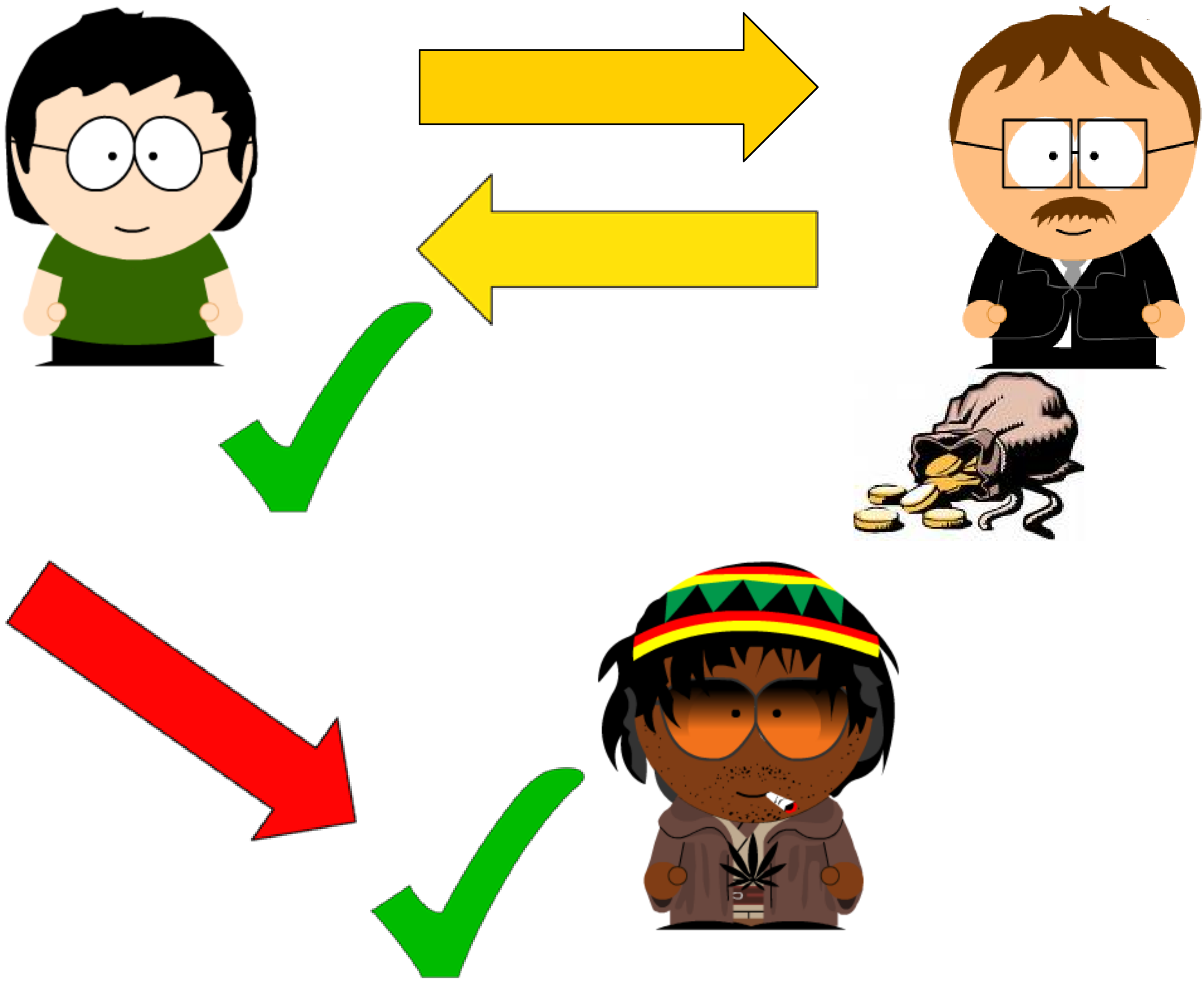
- Deposit($\mathcal{M}(sk_\mathcal{M}, S, \pi, pk_\mathcal{B}), \mathcal{B}(pk_\mathcal{M}, sk_\mathcal{B})$) protocol, a merchant $\mathcal{M}$ deposits a coin (S, π) into its account held by the bank $\mathcal{B}$. Whenever an honest $\mathcal{M}$ obtained (S, π) by running the Spend protocol with any (honest or otherwise) user, there is a guarantee that this coin will be accepted by the bank. $\mathcal{B}$ adds (S, π) to to its list L of spent coins. The merchants output is nothing or an error message.

E-Cash Protocol Spec

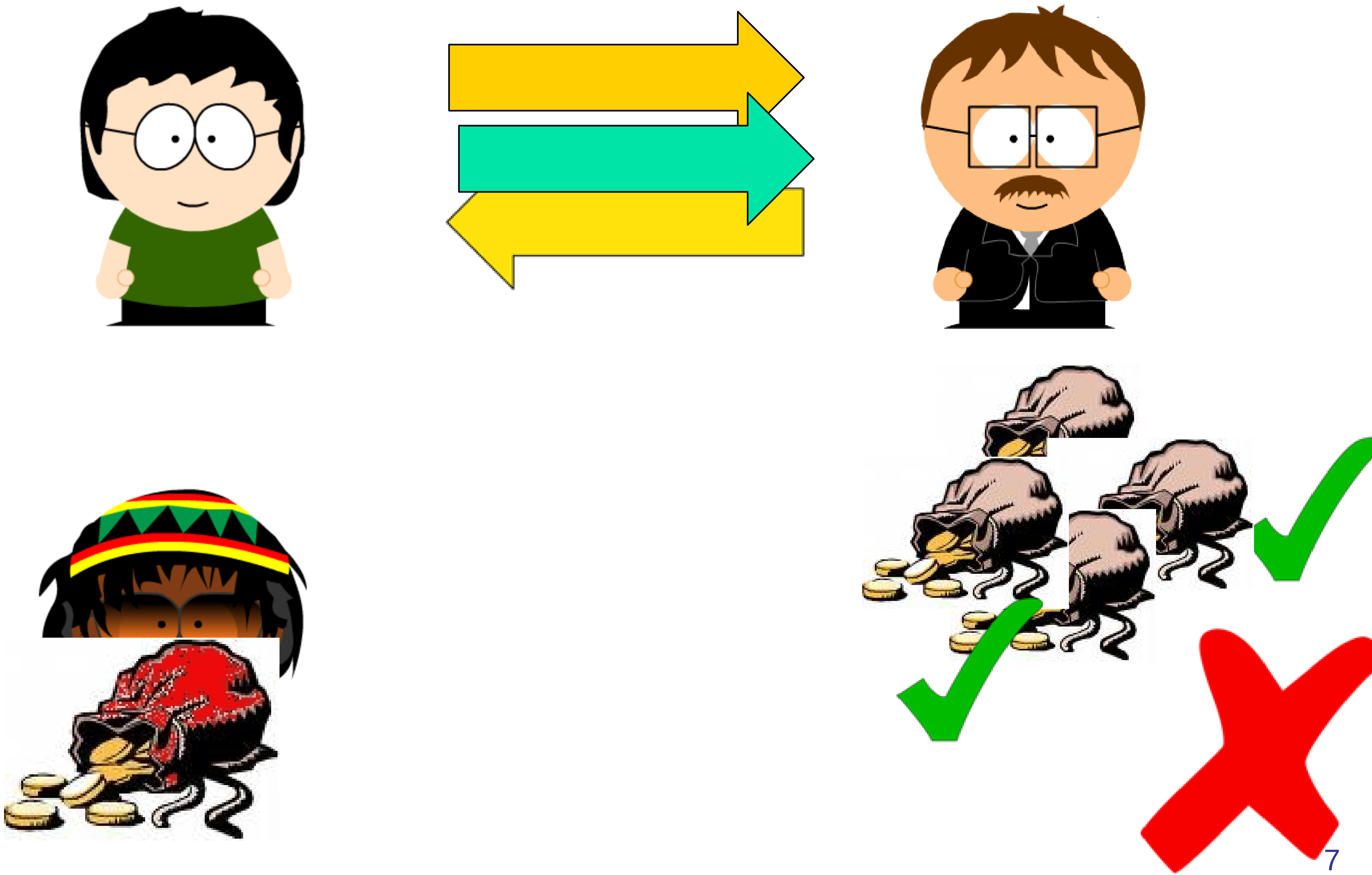
- $\mathsf{Identify}(params, S, \pi_1, \pi_2)$ algorithm allows to identify double-spenders using a serial number S and two proofs of validity of this coin, π_1 and π_2, possibly submitted by malicious merchants. This algorithm outputs a public key $pk_{\mathcal{U}}$ and a proof Π_G. If the merchants who had submitted π_1 and pi_2 are not malicious, then Pi_G is evidence that $pk_{\mathcal{U}}$ is the registered public key of a user that double-spent coin S

- $\mathsf{VerifyGuilt}(params, S, pk_{\mathcal{U}}, \Pi_G)$ algorithm allows to publicly verify proof Π_G that the user with public key $pk_{\mathcal{U}}$ is guilty of double-spending coin S.

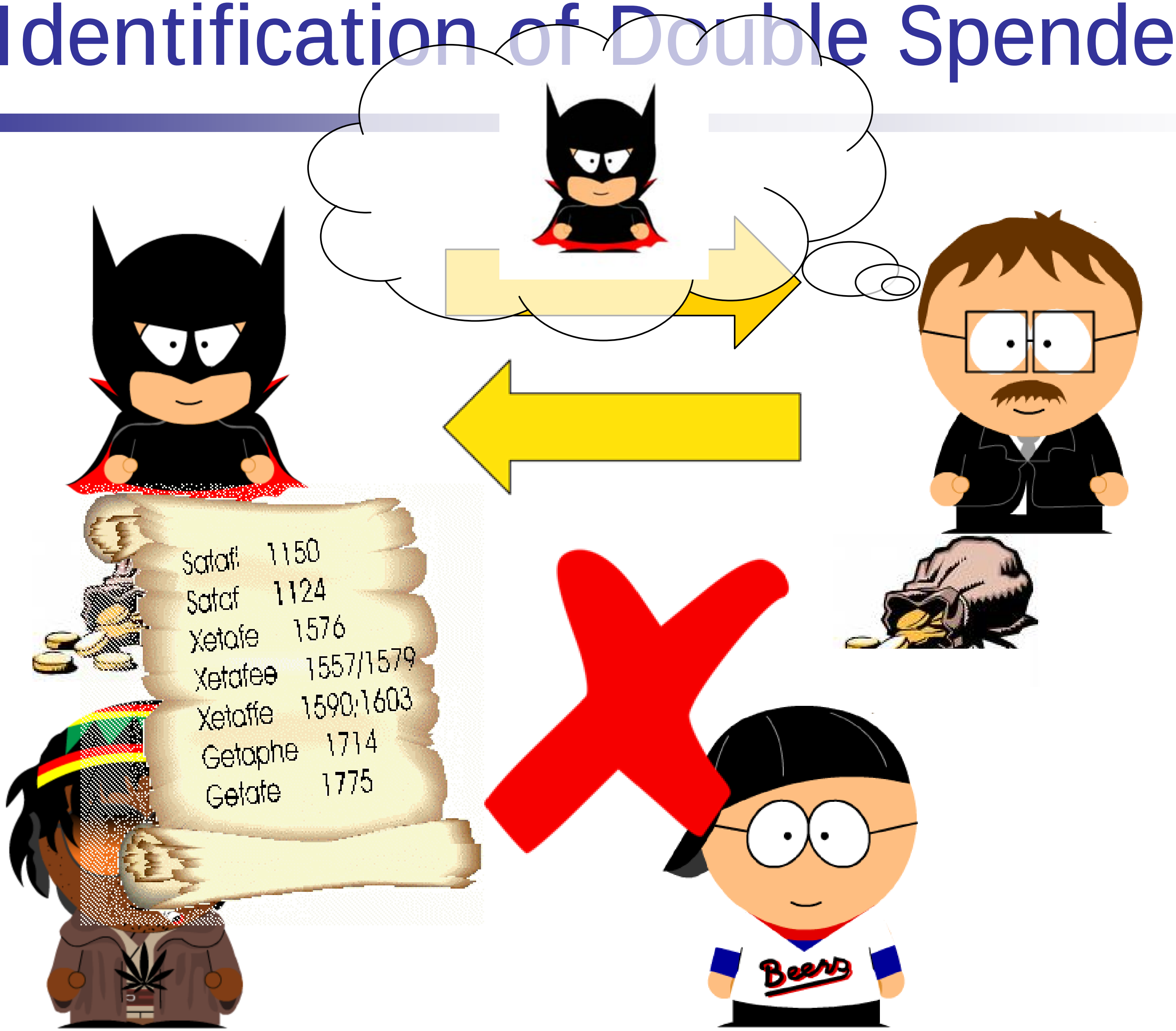
Correctness



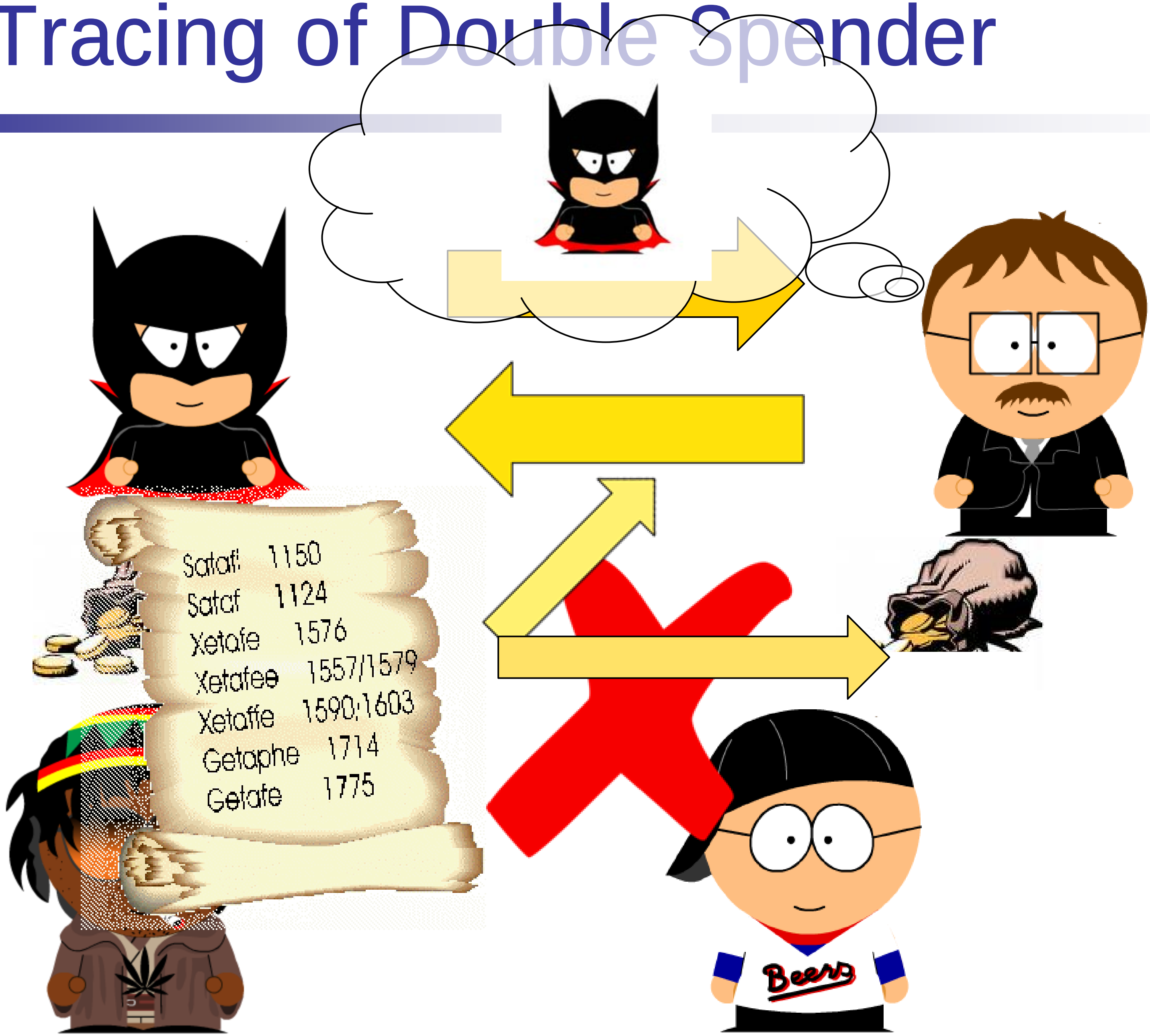
Balance



Identification of Double Spender



Tracing of Double Spender



Anonimity of the User

An E-Cash Protocol [CHL05]

Withdraw($\mathcal{U}(pk_{\mathcal{B}}, sk_{\mathcal{U}}, 2^{\ell}), \mathcal{B}(pk_{\mathcal{U}}, sk_{\mathcal{B}}, 2^{\ell})$): A user $\mathcal{U}$ interacts with the bank $\mathcal{B}$ as follows:

1. $\mathcal{U}$ identifies himself to the bank $\mathcal{B}$ by proving knowledge of $sk_{\mathcal{U}}$.
2. In this step, the user and bank contribute randomness to the wallet secret s; the user also selects a wallet secret t. This is done as follows: $\mathcal{U}$ selects random values $s', t \in \mathbb{Z}_q$ and sends a commitment $A' = \mathrm{PedCom}(u, s', t; r)$ to $\mathcal{B}$. $\mathcal{B}$ sends a random $r' \in \mathbb{Z}_q$. Then $\mathcal{U}$ sets $s = s' + r'$. $\mathcal{U}$ and $\mathcal{B}$ locally compute $A = g_2^{r'} A' = \mathrm{PedCom}(u, s' + r', t; r) = \mathrm{PedCom}(u, s, t; r)$.
3. $\mathcal{U}$ and $\mathcal{B}$ run the CL protocol for obtaining $\mathcal{B}$'s signature on committed values contained in commitment A. As a result, $\mathcal{U}$ obtains $\sigma_{\mathcal{B}}(u, s, t)$.
4. $\mathcal{U}$ saves the wallet $W = (sk_{\mathcal{U}}, s, t, \sigma_{B}(u, s, t), J)$, where s, t are the wallet secrets, $\sigma_{\mathcal{B}}(u, s, t)$ is the bank's signature, and J is an ℓ-bit coin counter initialized to zero.
5. $\mathcal{B}$ records a debit of 2^{ℓ} coins for account $pk_{\mathcal{U}}$.

An E-Cash Protocol [CHL05]

Spend($\mathcal{U}(W, pk_{\mathcal{M}}), \mathcal{M}(sk_{\mathcal{M}}, pk_{\mathcal{B}}, 2^{\ell})$): $\mathcal{U}$ anonymously transfers a coin to $\mathcal{M}$ as follows. (An optimized version appears in the full version of this paper [13].)

1. $\mathcal{M}$ (optionally) sends a string $info \in \{0,1\}^*$ containing transaction information to $\mathcal{U}$ and authenticates himself by proving knowledge of $sk_{\mathcal{M}}$.
2. $\mathcal{M}$ chooses a random $R \in \mathbb{Z}_q^*$ and sends R to $\mathcal{U}$. This is for the double-spending equation (see Section 1).
3. $\mathcal{U}$ sends to $\mathcal{M}$ the serial number of the coin $S = F_s^{DY}(J)$, and security tag $T = pk_{\mathcal{U}} F_t^{DY}(J)^R$. Now $\mathcal{U}$ must prove their validity, i.e., that S and T correspond to wallet secrets (u, s, t) signed by $\mathcal{B}$. This is done as follows:
   (a) Let $A = \mathrm{PedCom}(J)$; prove that A is a commitment to an integer in the range $[0 \ldots 2^{\ell} - 1]$.
   (b) Let $B = \mathrm{PedCom}(u)$, $C = \mathrm{PedCom}(s)$, $D = \mathrm{PedCom}(t)$; prove knowledge of a CL signature from $\mathcal{B}$ on the openings of B, C and D in that order,
   (c) Prove $S = F_s^{DY}(J) = \mathrm{g}_2^{1/(J+s+1)}$ and $T = pk_{\mathcal{U}} F_t^{DY}(J)^R = \mathrm{g}_2^{u+R/(J+t+1)}$. More formally, this proof is the following proof of knowledge:

$$PK\{(\alpha, \beta, \delta, \gamma_1, \ldots, \gamma_3) : g_1 = (AC)^{\alpha} {h_1}^{\gamma_1} \wedge S = \mathrm{g}_2^{\alpha} \wedge \\ g_1 = (AD)^{\beta} {h_1}^{\gamma_2} \ \wedge \ B = {g_1}^{\delta} {h_1}^{\gamma_3} \ \wedge \ T = \mathrm{g}_2^{\delta} (\mathrm{g}_2^{R})^{\beta}\}$$

   Use the Fiat-Shamir heuristic to turn all the proofs above into one signature of knowledge on the values $(S, T, A, B, C, D, g_1, h_1, n, \mathrm{g}_2, pk_{\mathcal{M}}, R, info)$. Call the resulting signature Φ.
4. If Φ verifies, $\mathcal{M}$ accepts the coin (S, π), where $\pi = (R, T, \Phi)$, and uses this information at deposit time.
5. $\mathcal{U}$ updates his counter $J = J + 1$. When $J > 2^{\ell} - 1$, the wallet is empty.

Problem

- How to symbolically model E-Cash Protocols?
- How to reason about it?
  - E-Cash properties involve elaborated cryptographic statements
- And what about reasoning on E-Cash at an application layer?
  - Do we need to redo all these proofs?
  - How is the interaction of programs and crypto?
- Are we willing to model every single cryptographic or do we rather prefer to model E-Cash primitives as BB?

The 3 Layer Cake

Well-behaved Semantics

Intermediate Semantics

Crypto

Users are modelled as applied pi-calculus processes

All users follow the specification

Double spending is prevented by construction

Environment is an arbitrary process

This Work

- We consider symbolic characterizations of Compact E-Cash protocols following the specification proposed by Camenisch, Hohenberger, and Lysyanskaya [CHL05]
- We design and implement a distributed (asynchronous) process calculus with high-level E-Cash primitives and communication (following ideas of [AF06])
  - Our calculus supports simple reasoning, based on labelled transitions and observational equivalence
- We consider two variants of the symbolic semantics
  - An abstract semantics that excludes any double spending (by design)
  - A more realistic intermediate semantics that accounts for the possibility of double spending (with reliable detection)
  - We show that any trace of the intermediate semantics can be captured by the "honest" semantics or an alert is issued

This Work

- We then consider a direct cryptographic implementation of high-level E-Cash primitives (Withdraw, Spend and Deposit following ideas of [AF06])
- Relate the intermediate semantics to the computational properties of the underlying E-cash protocol
- We obtain soundness and completeness for processes, in the presence of active adversaries
- We do not rely on DY abstractions of cryptographic primitives
  - Full abstraction for spi or applied pi calculus is too hard

High-Level Processes

Processes and Systems

P	$::=$		processes
	$\mid$	0	null process
	$\mid$	$P_1 \mid P_2$	deterministic parallel composition
	$\mid$	$\nu\, c\,.\, P$	restriction of names
	$\mid$	$u?(x).P$	receive on u
	$\mid$	$u!\langle M\rangle.P$	output M on u, M is not a channel name
	$\mid$	if $M = M'$ then P else P'	conditional
	$\mid$	repl P	replication of a blocking process P
	$\mid$	withdraw! $u\ \mathcal{L}$	withdraw a coin from bank u
	$\mid$	withdraw? $u\ P$	create a coin for user u
	$\mid$	spend! $u\ u'\ p$	spend the coin u' to pay u through bank p
	$\mid$	spend? $p\ \mathcal{L}$	wait a payment through bank p
	$\mid$	deposit! $p\ u\ u'$	deposit a spent coin u to bank p (user u' is hidden)
	$\mid$	deposit? $p\ (x)\ P_1\ P_2$	bind a coin, and its depositor in P_1 if it is honest

$A,\ E$	$::=$		
	$\mid$	0	
	$\mid$	$p@P$	process owned by p
	$\mid$	$A_1 \mid A_2$	parallel composition
	$\mid$	$\nu\, u\,.\, A$	restriction of name

Reduction Semantics

$$U@\mathsf{withdraw!}\, B\, \mathcal{L} \mid B@\mathsf{withdraw?}\, U\, P \rightarrow_a \nu\, b\, .\, (U@\mathcal{L}[\mathsf{spend!}\, x\, b\, B] \mid B@P \mid \mathsf{coin}\, b\, B\,)$$

$$U@\mathsf{spend!}\, M\, b\, B \mid M@\mathsf{spend?}\, B\, \mathcal{L} \rightarrow_a M@\mathcal{L}[\mathsf{deposit!}\, B\, b\,]$$

$$M@\mathsf{deposit!}\, B\, b \mid B@\mathsf{deposit?}\, M\, (x)\, P_1\, P_2 \mid \mathsf{coin}\, b\, B \rightarrow_a B@P_1\{^{M}/_{x}\}$$

Labelled Transition Semantics

$$\frac{A \rightarrow A'}{A \xrightarrow{\tau} A'} \quad \textsc{AbsLtSilent}$$

$$\frac{}{a@c!\langle M\rangle.P \xrightarrow{c\,!M} a@P} \quad \textsc{AbsLtSend\_term}$$

$$\frac{}{a@c?(x).P \xrightarrow{c\,?M} a@P\{^{M}/_{x}\}} \quad \textsc{AbsLtReceive\_term}$$

$$\frac{\begin{array}{l} A \xrightarrow{c\,!M} A' \\ (\, c \neq r \,\wedge\, r \in M) \end{array}}{\nu\, r\,.\, A \xrightarrow{\nu r.c\,!M} A'} \quad \textsc{AbsLtOpen}$$

$$\frac{A \xrightarrow{\phi} A' \,\wedge\, c \notin \phi}{\nu\, c\,.\, A \xrightarrow{\phi} \nu\, c\,.\, A'} \quad \textsc{AbsLtScope}$$

$$\frac{A_1 \xrightarrow{\phi} A_1' \,\wedge\, BN(\phi) \cap FN(A_2) = \emptyset}{A_1 \mid A_2 \xrightarrow{\phi} A_1' \mid A_2} \quad \textsc{AbsLtPar}$$

$$\frac{A_1 \equiv A_2 \,\wedge\, (\, A_2 \xrightarrow{\phi} A_3 \,\wedge\, A_3 \equiv A_4\,)}{A_1 \xrightarrow{\phi} A_4} \quad \textsc{AbsLtLab\_struct}$$

Labelled Transition Semantics

$$U@\mathsf{withdraw!}\ B\ \mathcal{L} \xrightarrow{\mathsf{withdraw!}\ U\ B}_a \nu\, b\,.\, U@\mathcal{L}[\mathsf{spend!}\ x\ b\ B]$$

$$B@\mathsf{withdraw?}\ U\ P \xrightarrow{\mathsf{withdraw?}\ U\ B}_a B@P \mid \mathsf{coin}\ b\ B$$

$$U@\mathsf{spend!}\ M\ b\ B \xrightarrow{\mathsf{spend!}\ b\ B\ M}_a U@0$$

$$M@\mathsf{spend?}\ B\ \mathcal{L} \mid \mathsf{coin}\ b\ B \xrightarrow{\mathsf{spend?}\, b\ B\ M}_a M@\mathcal{L}[\mathsf{deposit!}\ B\ b\,] \mid \mathsf{coin}\ b\ B \qquad (\, B \in \mathcal{H}\,)$$

$$M@\mathsf{spend?}\ B\ \mathcal{L} \xrightarrow{\mathsf{spend?}\, b\ B\ M}_a M@\mathcal{L}[\mathsf{deposit!}\ B\ b\,] \qquad (\, B \notin \mathcal{H}\,)$$

$$M@\mathsf{deposit!}\ B\ b \xrightarrow{\mathsf{deposit!}\ b\ B\ M}_a M@0$$

$$B@\mathsf{deposit?}\ M\ (x)\ P_1\ P_2 \mid \mathsf{coin}\ b\ B \xrightarrow{\mathsf{deposit?}\ b\ B\ M}_a B@P_1\{^M/_x\}$$

Intermediate Reduction Semantics

$U@\text{withdraw!}\ B\ \mathcal{L}\ \mid\ B@\text{withdraw?}\ U\ P \longrightarrow_i \nu\, b\ \boxed{(U@\mathcal{L}[\text{spend!}\ x\ b\ B]}\ \mid\ B@P\ \mid \boxed{\text{coin}\ b\ B\ U)}$

$U@\text{spend!}\ M\ b\ B\ \mid\ M@\text{spend?}\ B\ \mathcal{L} \longrightarrow_i M@\mathcal{L}[\text{deposit!}\ B\ b\,]$

$M@\text{deposit!}\ B\ b\ \mid\ B@\text{deposit?}\ M\ (x)\ P_1\ P_2\ \mid\ \text{coin}\ b\ B\ U \longrightarrow_i B@P_1\{^M/_x\}\ \mid \boxed{\text{coin}^{\text{d}}\ b\ B\ U}$

$\boxed{M@\text{deposit!}\ B\ b\ \mid\ B@\text{deposit?}\ M\ (x)\ P_1\ P_2\ \mid\ \text{coin}^{\text{d}}\ b\ B\ U \longrightarrow_i B@P_2\{^{\mathbf{bad}(b\,,\,U)}/_x\}\ \mid\ \text{coin}^{\text{d}}\ b\ B\ U}$

Intermediate LT Semantics

$$U@\text{withdraw!}\ B\ \mathcal{L} \xrightarrow{\text{withdraw!}\ \ U\ B}_i \nu\, b\ \boxed{U@\mathcal{L}[\text{spend!}\ x\ b\ B]}$$

$$B@\text{withdraw?}\ U\ P \xrightarrow{\text{withdraw?}\ \ U\ B}_i B@P \mid \text{coin}\ b\ B\ U$$

$$U@\text{spend!}\ M\ b\ B \xrightarrow{\text{spend!}\ b\ B\ M}_i U@0$$

$$M@\text{spend?}\ B\ \mathcal{L} \mid \boxed{\text{coin}^*\ b\ B\ U} \xrightarrow{\text{spend?}\ b\ B\ M}_i M@\mathcal{L}[\text{deposit!}\ B\ b] \mid \boxed{\text{coin}^*\ b\ B\ U}\ (B \in \mathcal{H})$$

$$M@\text{spend?}\ B\ \mathcal{L} \xrightarrow{\text{spend?}\ b\ B\ M\ \ U}_i M@\mathcal{L}[\text{deposit!}\ B\ b] \qquad (B \notin \mathcal{H})$$

$$M@\text{deposit!}\ B\ b \xrightarrow{\text{deposit!}\ b\ B\ M}_i M@0$$

$$B@\text{deposit?}\ M\ (x)\ P_1\ P_2 \mid \text{coin}\ b\ B\ U \xrightarrow{\text{deposit?}\ b\ B\ M}_i B@P_1\{^M/_x\} \mid \boxed{\text{coin}^{\text{d}}\ b\ B\ U}$$

$$\boxed{B@\text{deposit?}\ M\ (x)\ P_1\ P_2 \mid \text{coin}^{\text{d}}\ b\ B\ U \xrightarrow{\text{deposit?}\ b\ B\ M}_i B@P_2\{^{\mathbf{bad}(b\ ,\ U)}/_x\} \mid \text{coin}^{\text{d}}\ b\ B\ U}$$

High-Level Reasoning

Example – Properties

- *Correctness.* An honest user can withdraw a coin from an honest bank, according to WITHDRAW.

  Only the user specified by the bank can receive the spend continuation.

  Only the merchant specified by the client can receive the deposit continuation.

- *Balance.* For any series of reductions of an initial high level configuration, rule DEPOSIT cannot be applied more often than WITHDRAW

- *Anonymity of users.* A bank cannot learn anything about a user's spendings

$$p_1@\mathsf{spend!}\, b\ S\ B \mid p_2@0 \approx p_1@0 \mid p_2@\mathsf{spend!}\, b\ S\ B \quad \forall p_1, p_2$$

Example – Properties

- *Identification of double-spenders.* Given two records of a double spent coin, the identity of the user can be extracted
- *Weak Exculpability.* Only double-spenders can be accused
- *Strong Exculpability.* an user can only be responsible for coins that he indeed double-spent (each $\mathbf{bad}(b\ ,\ U)$ binds user U to a particular coin b).

Soundness, Completeness for High-Level Semantics

Main Results

Well-behaved Semantics

Intermediate Semantics

Crypto

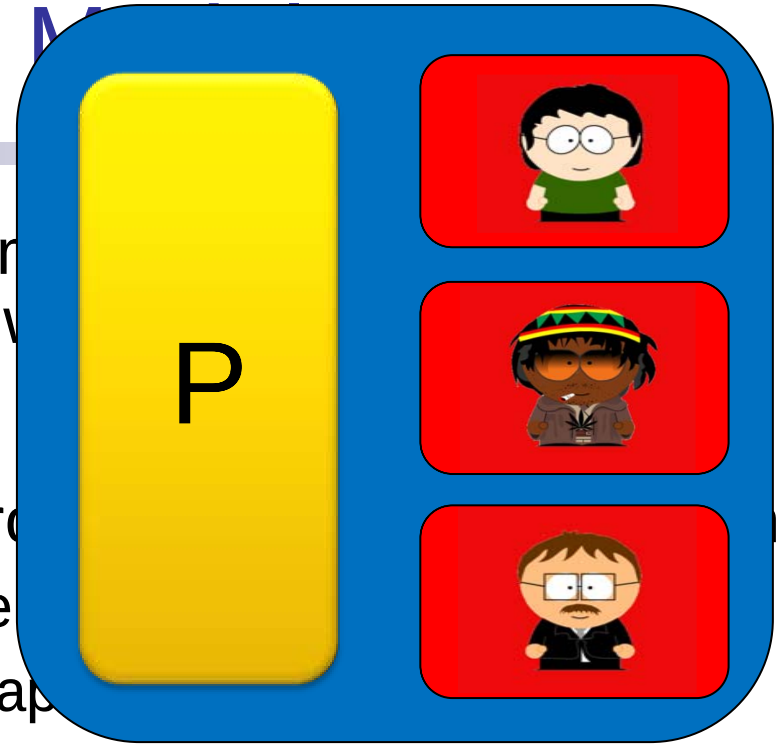
Claim 1 (Relative correctness) *If two high level processes are equivalent then they are also equivalent in the weak intermediate semantics, if* $A \sim A'$ *then* $A \sim_w A'$.

Claim 2 (Completeness) *If two well-formed high level processes are equivalent in the intermediate level semantics then they are also equivalent in the high level semantics, if* $A \sim_i A'$ *then* $A \sim A'$.

Claim 3 (Progress) *If* P *is cheated at bank* B *on coin* b*, and if* B *is honest, then spender of the coin will be accused by* B.

Low-Level Target Model

Low-Level Excution

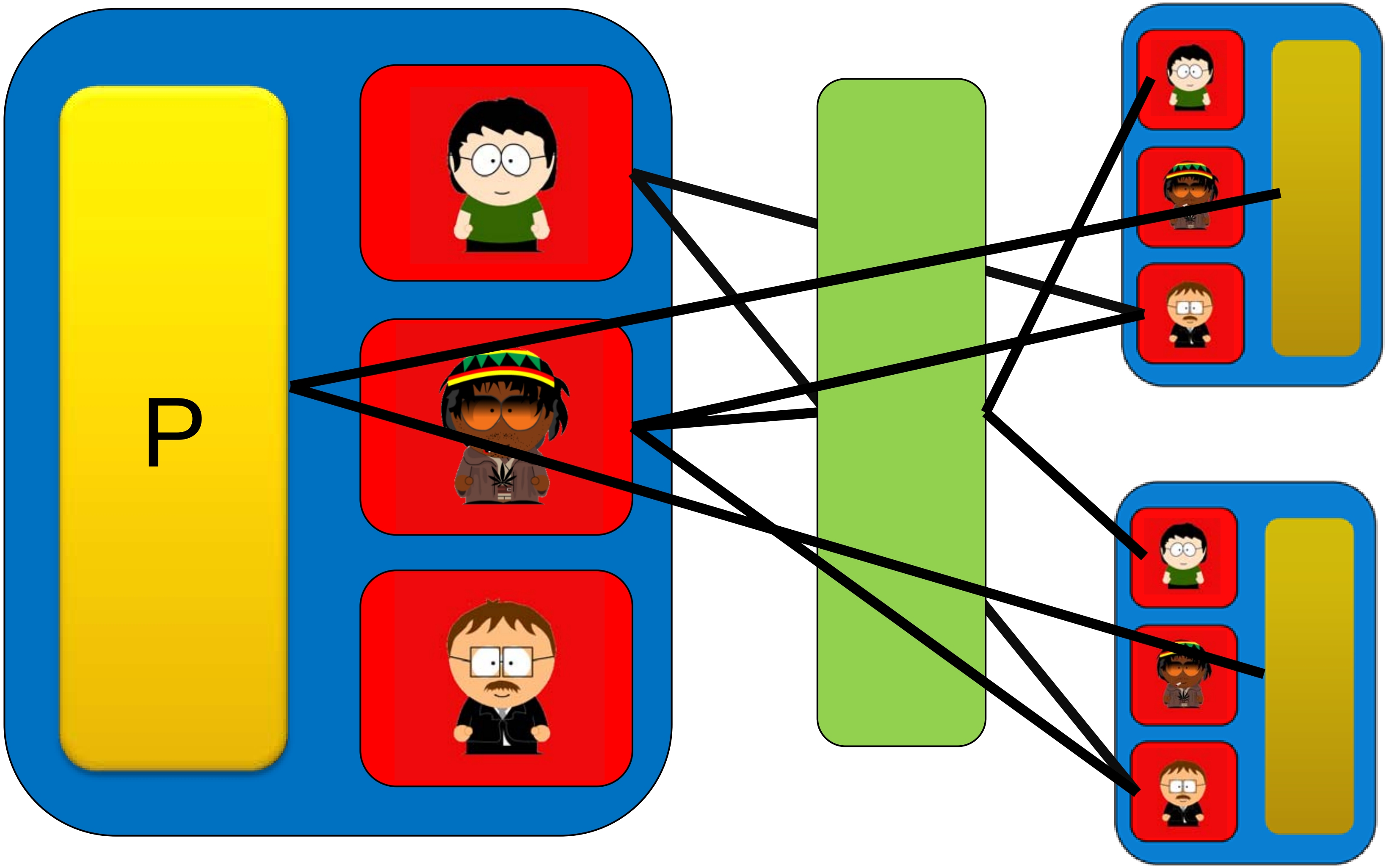


- Systems consist of a finite nu
  principals that are complying
  - THEY MAY DOUBLE SPEND
- Each principal runs its own pr ... e
  - includes 3 crypto boxes to pe
  - has an input and an output ta
- Priority is given to honest users
- Whenever all machines complete, all messages to the adversary are shuffled, and given to the adversary

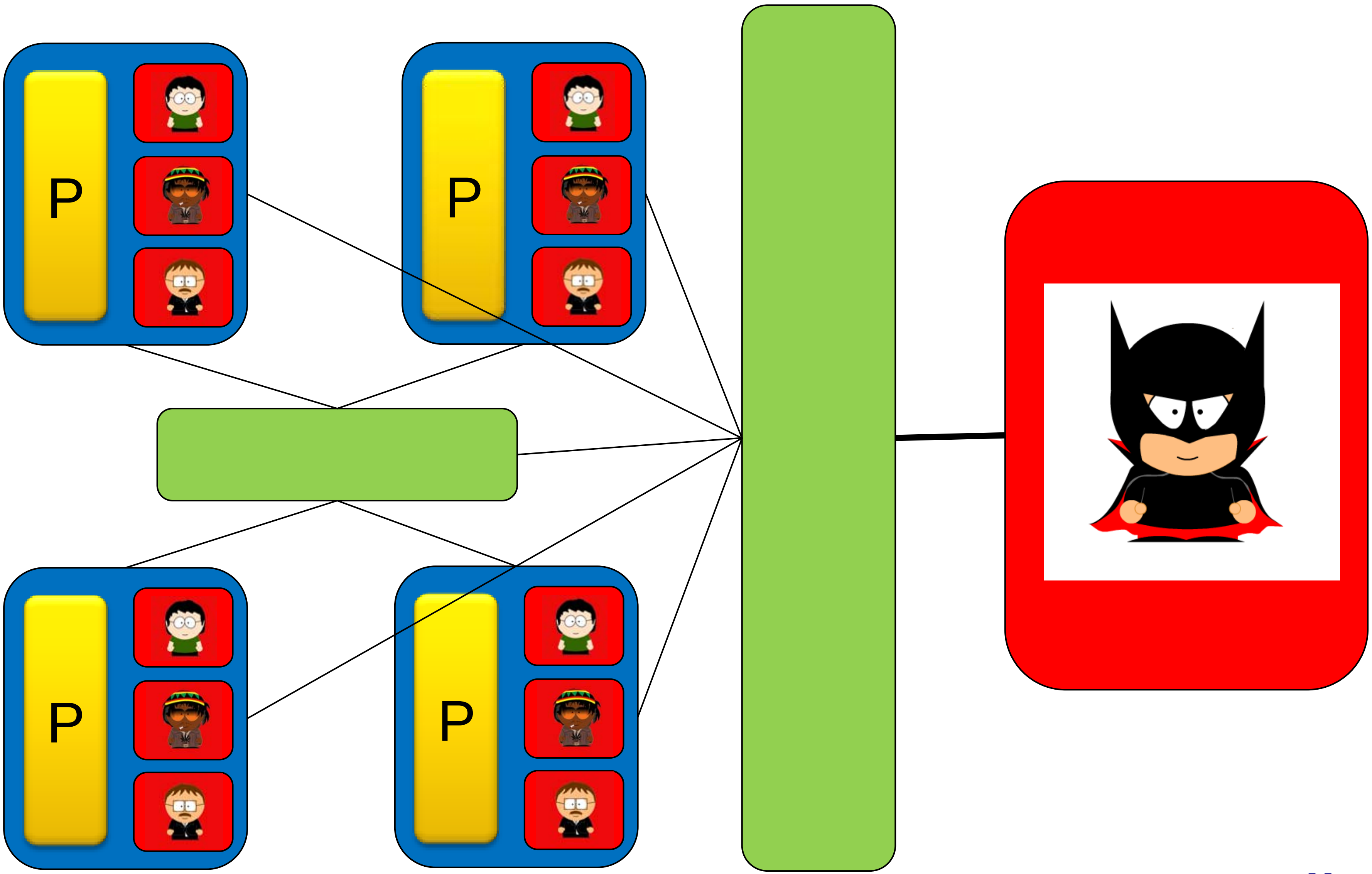
Low-Level Execution Model

- When activated, a machine reads one message from its input tape
  - an ecash message is routed to the appropriate cryptobox
  - a communication message, is sent to the runing process
- Ecash primitives in evaluation context
  - An output primitive triggers a new session of the protocol
  - An input primitive starts a waiting thread
- All machines should run to completion
  - consume all messages in its input tape and
  - write all output messages in the input tapes of receivers

Low Level Communications Model

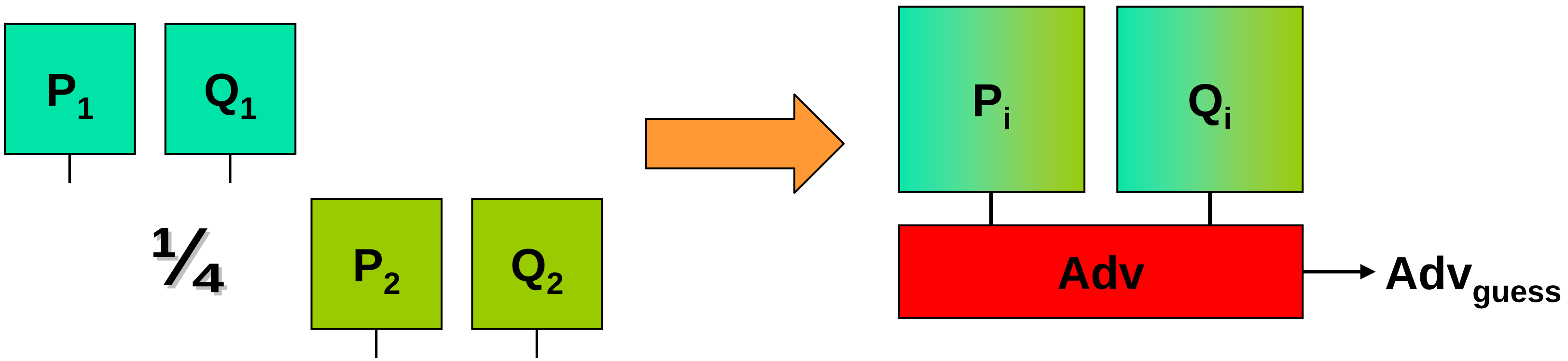


Low Level Communications Model



Low-Level Equivalence (Target)

Two PPT systems M^0 and M^1 are equivalent, written $M^0 \approx M^1$, when for every PPT adversary A, we have $|\Pr[1 \longleftarrow A[M^0]_\eta] - \Pr[1 \longleftarrow A[M^1]_\eta]| \leq \text{neg}(\eta)$.



Soundness, Completeness wrt Computational Cryptography

Main Results

Theorem 1 (Correctness) *For all systems A with a single evaluation context and* D *if* $A \xrightarrow{\phi}_a A'$ *then* $\exists \mathsf{A}, \tilde{\sigma}, \mathsf{D}'$ *such that* $\mathsf{A}[\mathsf{M}(A, \mathsf{D})] \overset{\tilde{\sigma}}{\rightsquigarrow} \mathsf{M}(A', \mathsf{D}')$.

Theorem 2 (Completeness) *If the implementations of two intermediate processes are equivalent then those processes are also equivalent.*

Claim 3 (Trace Lifting) *For all systems A and* D, *if* $\mathsf{M}(A, \mathsf{D}) \overset{\sigma_1 \sigma_2}{\rightsquigarrow} \mathsf{M}'$ *then* $\exists A, \mathsf{D}'$ *such that* $A \xrightarrow{\phi}_i A'$ *and* $\mathsf{M}' = \mathsf{M}(A', \mathsf{D}')$.

Claim 4 (Soundness) *If two intermediate processes are equivalent then with an overwhelming probability their low level implementations are also equivalent.*

Well-behaved Semantics

Intermediate Semantics

Crypto

Summary

- Reasoning about cryptographic actions is much more fun than cryptographic terms ☺
- We define a 3-layer cake to reason about E-Cash protocols
  - A low Crypto layer
  - An intermediate symbolic layer where probabilities are discarded
  - A higher symbolic layer where bad behaviours do not occur (by construction)
- We “show” that the low-level crypto layer is correctly abstracted by the “well-behaved” semantics
- Formalizing properties of E-Cash protocols helped us understand and fix part of the specification

Future Work

- Since it is work in progress….
- Try to debug and clarify specification
- Try to adapt similar techniques to other instances
  - Eg, e-voting

CoSyProofs, April6-9. 2009

High-Level Programming for E-Cash

Pedro Adão
IT and IST, Lisboa

Cédric Fournet
Microsoft Research and
MSR-INRIA Joint Centre

Nataliya Guts
MSR-INRIA Joint Centre

Francesco Zappa Nardelli
INRIA and
MSR-INRIA Joint Centre