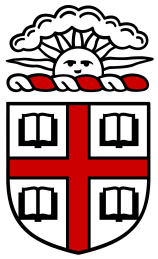
P4R: Privacy-Preserving Pre-Payments with Refunds for Transportation Systems

Andy Rupp[1], Gesine Hinterwälder[2], Foteini Baldimtsi[3], Christof Paar[2,4]

[1] Karlsruhe Institute of Technology
[2] University of Massachusetts Amherst
[3] Brown University
[4] Ruhr-University Bochum



0964641

Outline

- Motivation
- eCash
  - Overview
  - Performance Issues
- P4R
  - Description
  - Evaluation

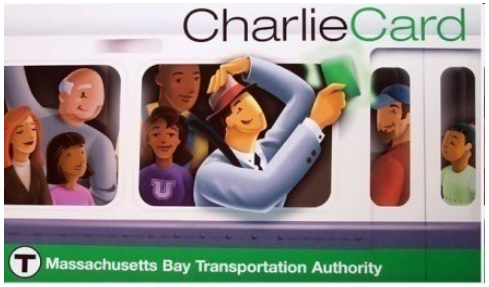
Motivation

- Transportation Payments
  - Large volumes
  - Low cost
  - Have to be executed fast
- Electronic Payments
  - Throughput and convenience advantages
  - Reduced revenue collection cost
  - Enable dynamic pricing
  - Facilitate maintenance of a system
  - Enable easy collection of meaningful data

Motivation

"Hacking the T: MBTA sues to keep MIT students from telling how they cracked the CharlieCard"

"Hackers Crack London Tube Oyster Card"



"Some call T's new Charlie Card an invasion of privacy. But agency insists safeguards in place"

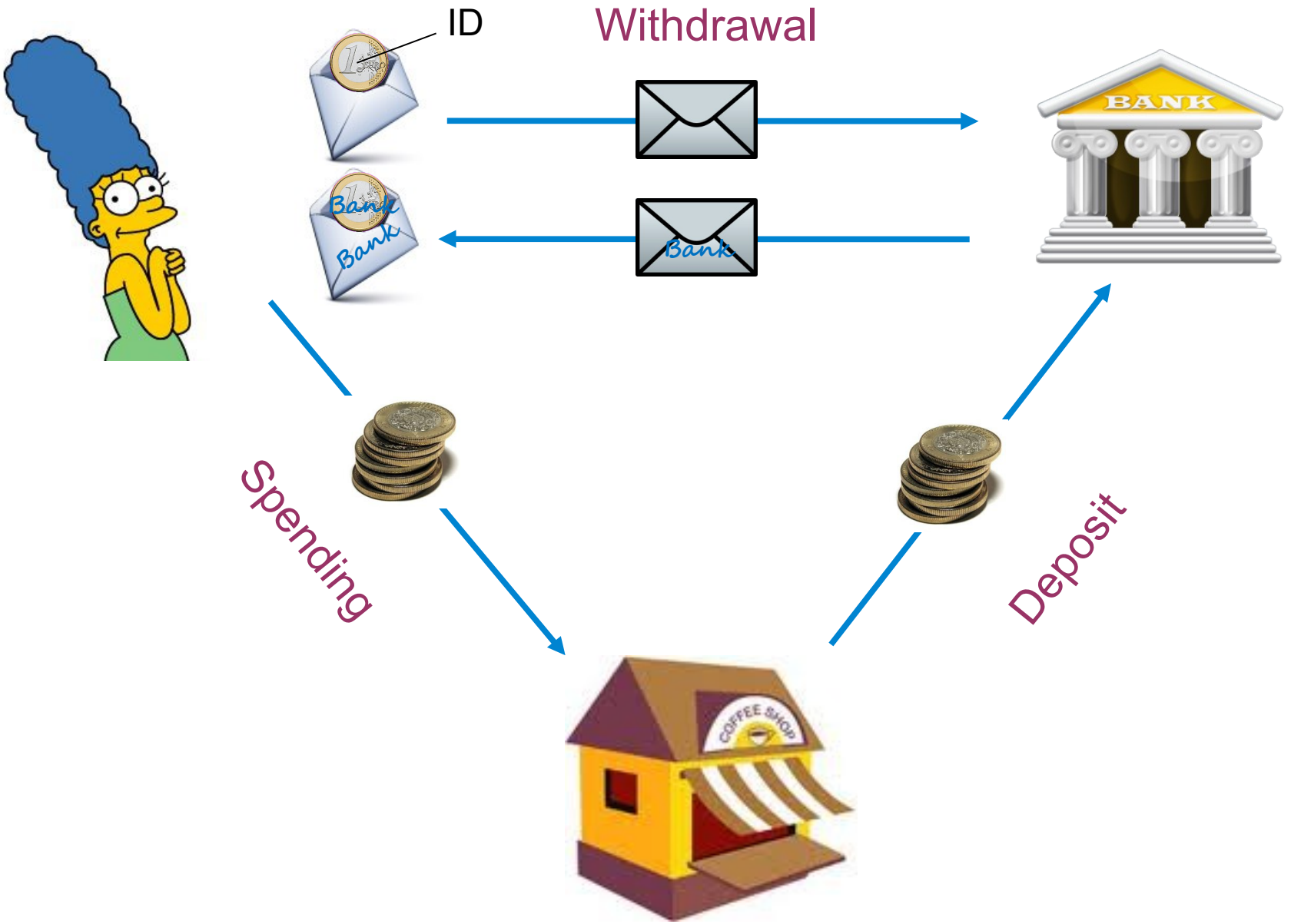
"Privacy Concerns Raised Over Clipper Card Passenger Tracking"

Motivation

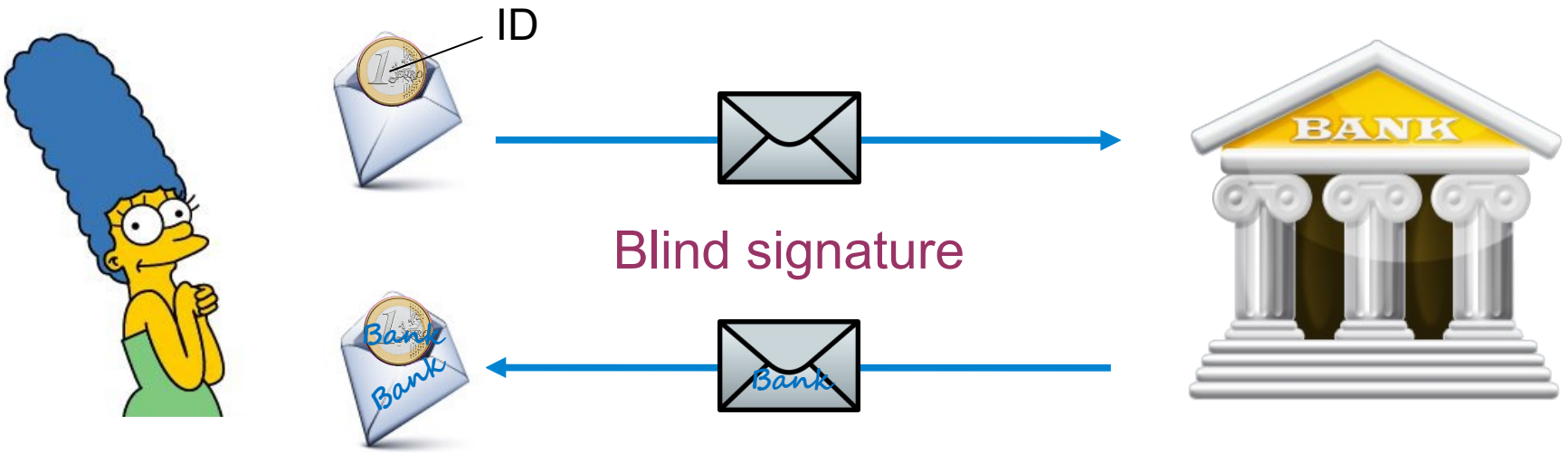
We need payment systems for transportation that are:

- Secure (unforgeable & secure against doublespending)
- Private (anonymous)
- Trusted
- Efficient
- Low-cost
- Usable
- Reliable

eCash



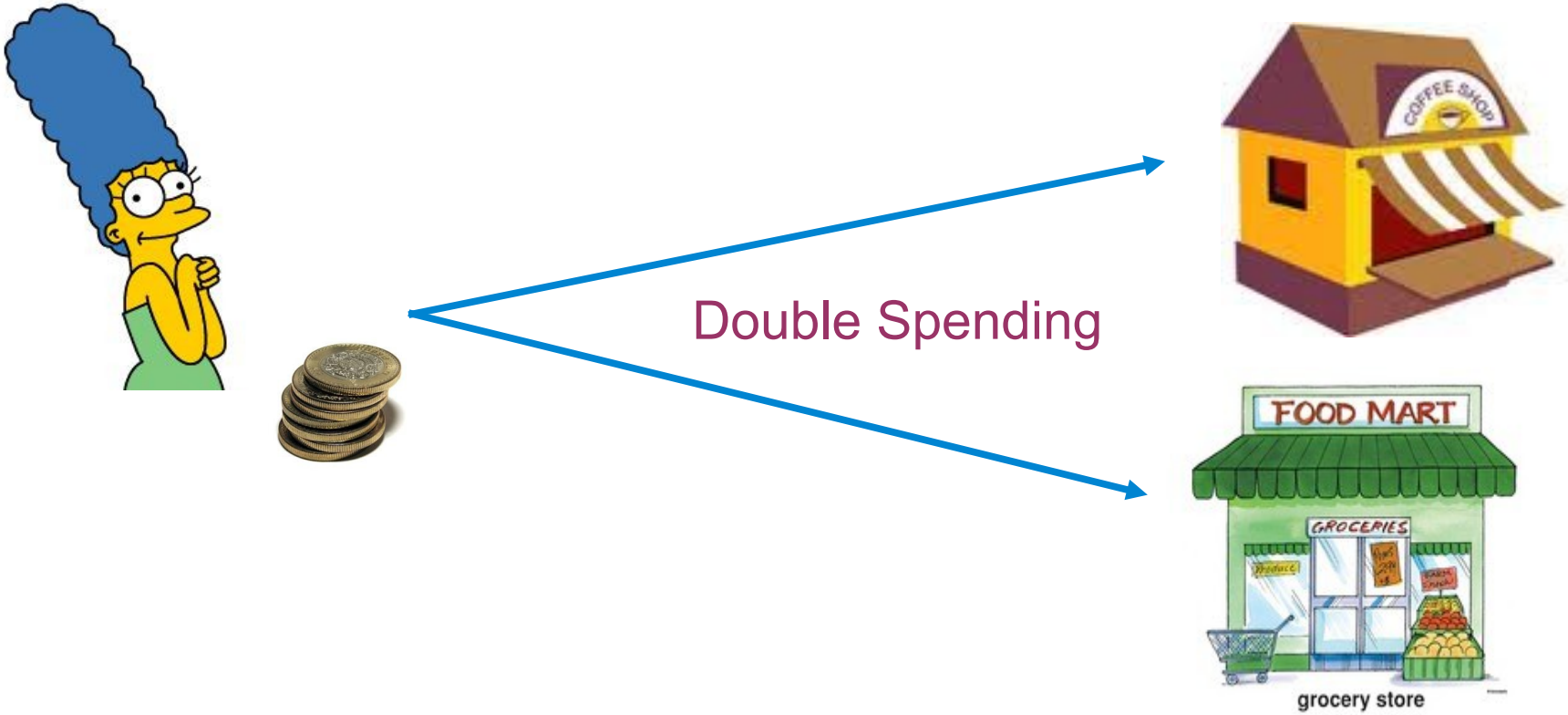
eCash



Security Properties of Blind Signatures

- Blindness: Signer should not be able to view the messages he signs (i.e. Bank cannot link e-coins to specific users)
- Unforgeability: User should not be able to forge the signer's signatures (i.e. User cannot forge coins)

eCash



Double Spending reveals User's ID!!!

Brands' Untraceable Offline Cash

- Introduced in 1993
- Most efficient scheme during Spending Phase
- Well-known and implemented (Microsoft U-Prove)

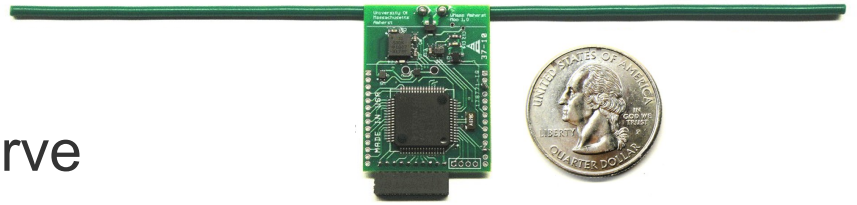
[Bra93] S. Brands. Untraceable Off-line Cash in Wallets with Observers (Extended Abstract). In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '93, pages 302–318, 1994.*

Brands' Untraceable Offline Cash

- Scheme based on cyclic group G_q of prime order
- Coin size (elements that have to be stored on user device for each coin): $A, B, z', a', b' \in G_q$ and $r', s, x_{0,} x_1 \in \mathbb{Z}_q$

		
Withdrawal	12 exponentiations	2 exponentiations
Spending		
	0 exponentiations	3 exponentiations

Implementation Results Brands'



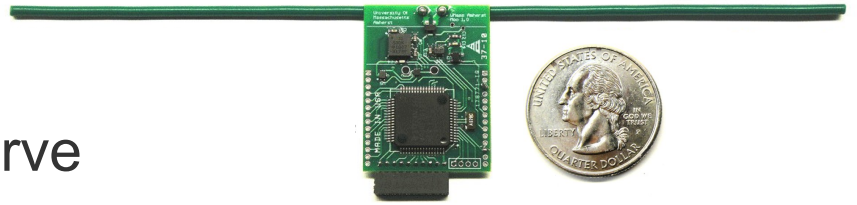
- Base scheme on 160-bit elliptic curve

  and measure execution time on Moo computational RFID tag
- Storage space required per coin: 284 bytes
- Execution time on MSP430F2618, when based on 160-bit curve:

	Cycle count	Execution time @16 MHz
Brands' withdrawing one coin	69 120 181	4.32 s
Brands' spending one coin	35 052	0.0022 s

[ZGRF11] H. Zhang, J. Gummeson, B. Ransford, and K. Fu. Moo: A Batteryless Computational RFID and Sensing Platform. https://web.cs.umass.edu/publication/docs/2011/UM-CS-2011-020.pdf. 2011.

Implementation Results Brands'



- Base scheme on 160-bit elliptic curve

  and measure execution time on Moo computational RFID tag
- Storage space required per coin: 284 bytes
-

Users should not have to withdraw and store too many coins!!!

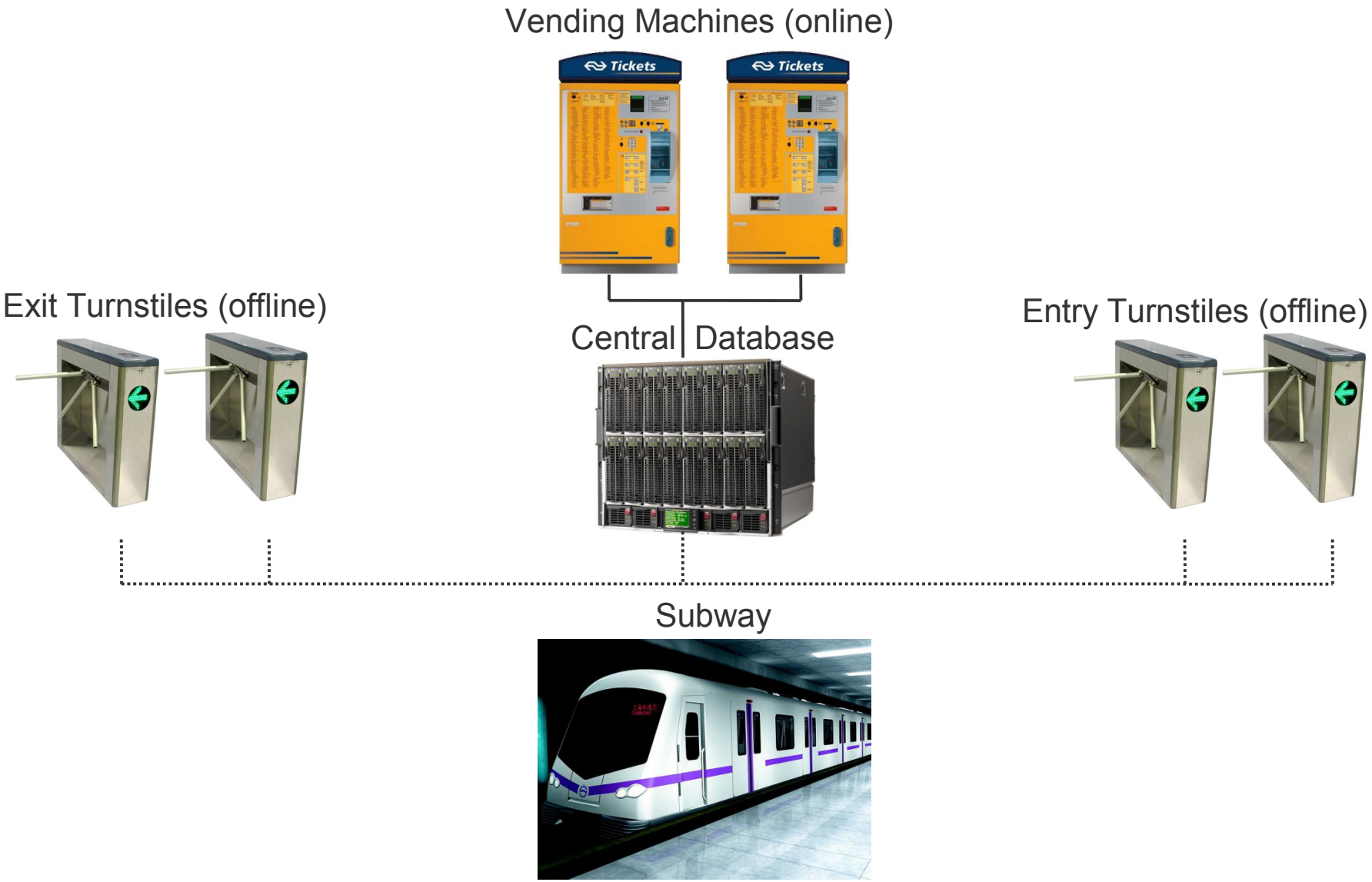
Brands' withdrawing one coin	69 120 181	4.32 s
Brands' spending one coin	35 052	0.0022 s

[ZGRF11] H. Zhang, J. Gummeson, B. Ransford, and K. Fu. Moo: A Batteryless Computational RFID and Sensing Platform. https://web.cs.umass.edu/publication/docs/2011/UM-CS-2011-020.pdf. 2011.

Our Approach

- **Build on Brands'** due to efficiency reasons (could use any efficient, anonymous 2-show credential scheme)
- **Alleviate its disadvantages** (large coin size, inefficient withdrawal)
- **Minimize number of coins needed** using novel *pre-payments with refunds* approach:
  - Use Brands' coin as ticket
  - Ticket price = cost of most expensive trip
  - Cost of actual trip determined on exit
  - Pay refund based on overpayment

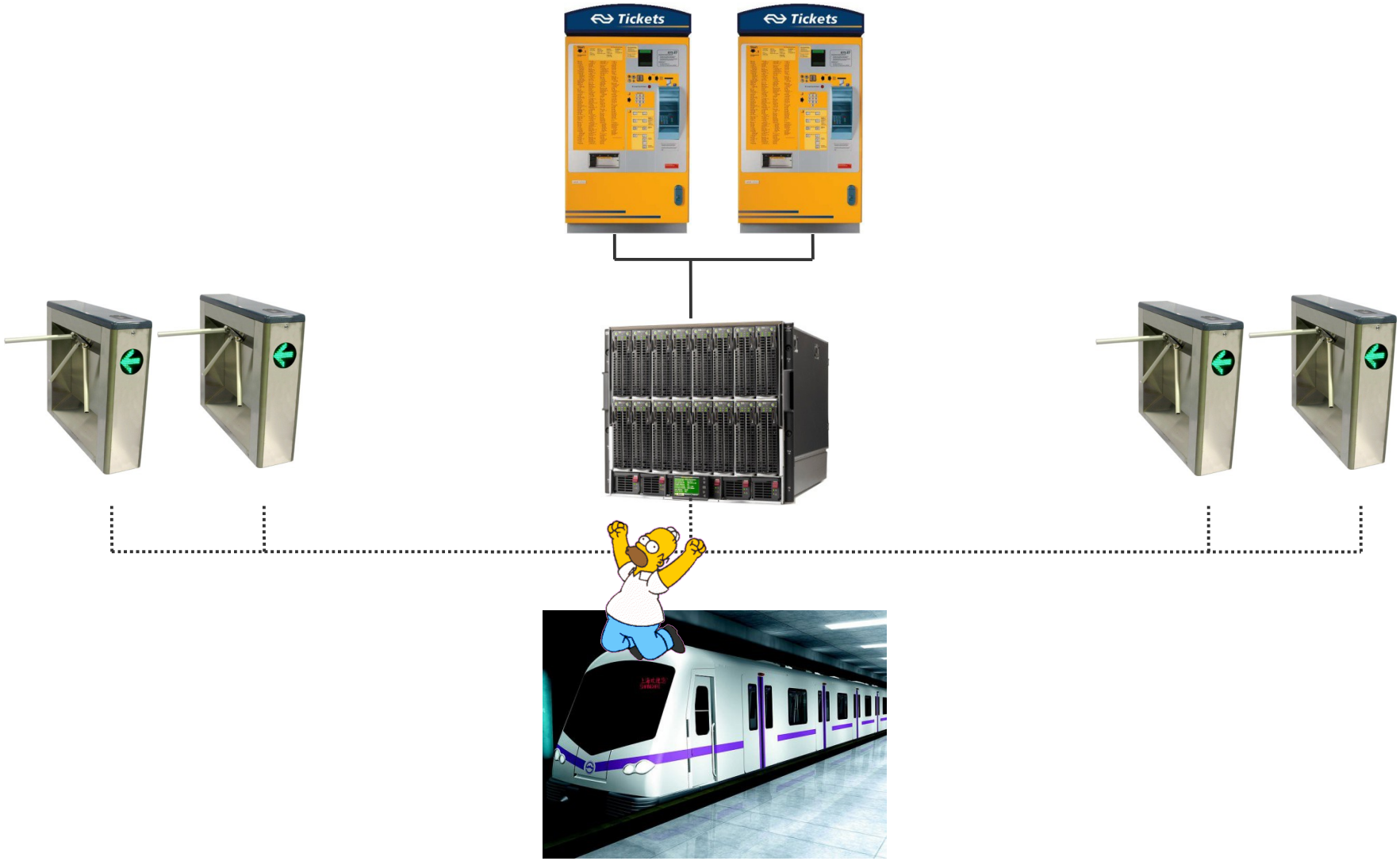
P4R: Main Components



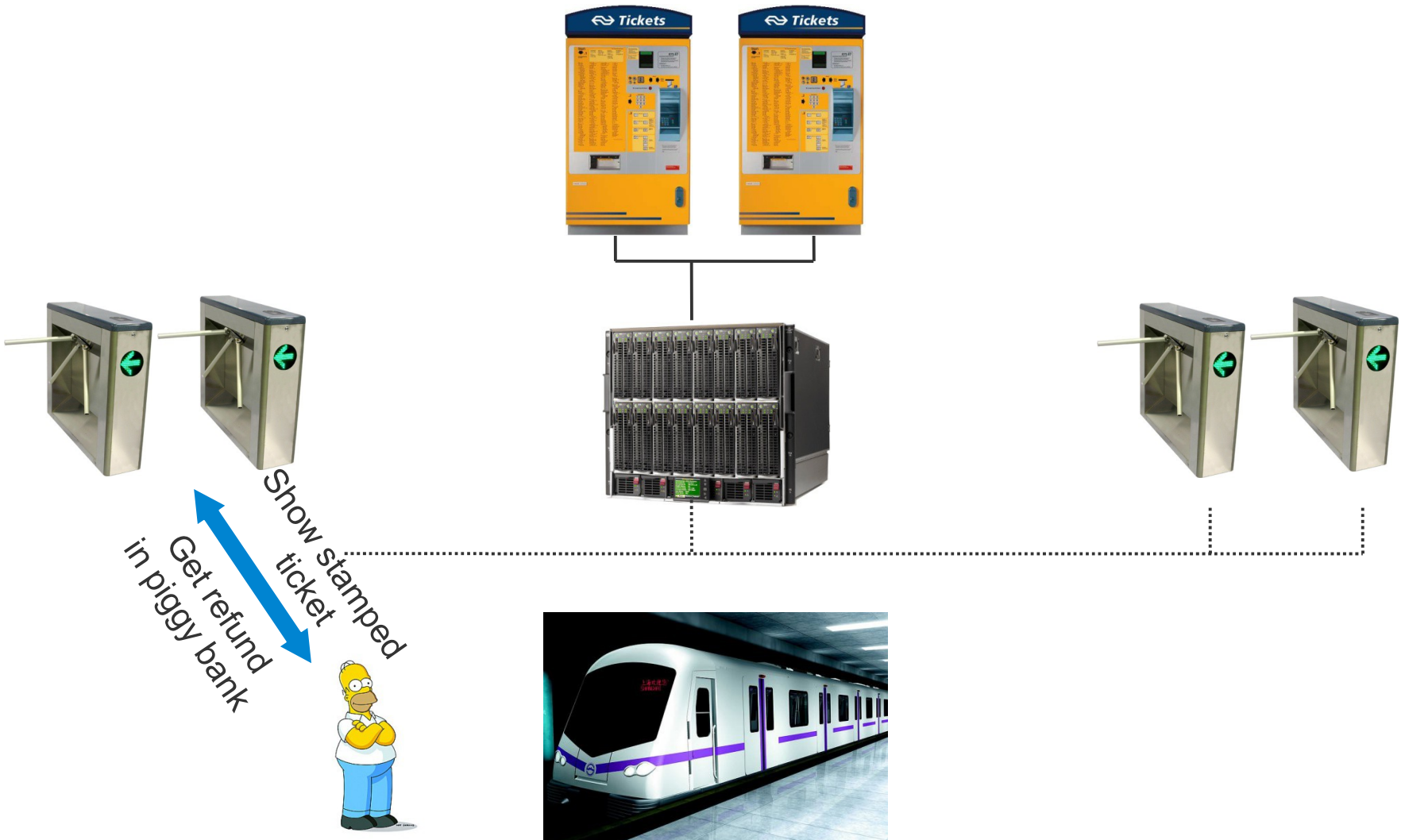
P4R: Main Components

P4R: Main Components

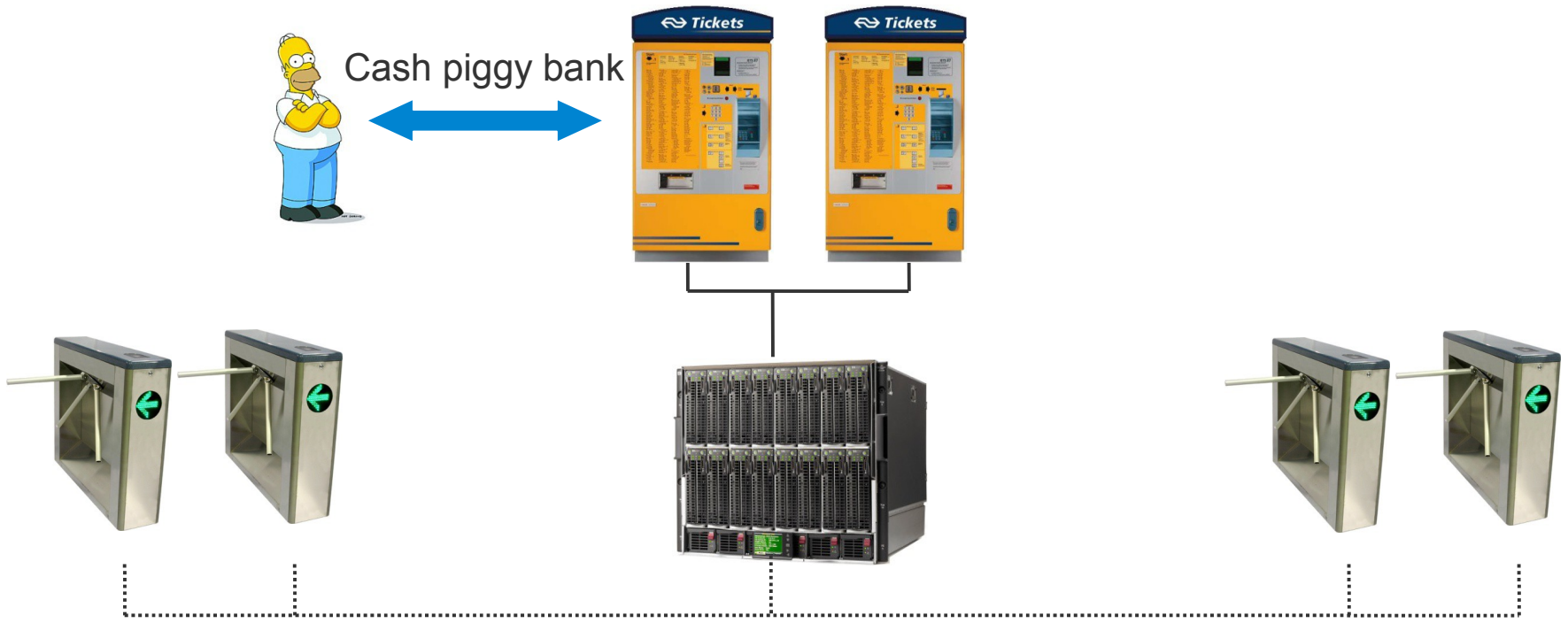
P4R: Main Components



P4R: Main Components



P4R: Main Components



Brands-Based TAT System

Brands' coin:

$$A = \left(g_1^{id_U} g_2\right)^s$$

$$B = g_1^{x_1} g_2^{x_2}$$

$$A, B, sig\left(A, B\right)$$

Showing coin:

$$r_1 = d\left(id_U s\right) + x_1$$

$$r_2 = d * s + x_2$$

Brands-Based TAT System

Brands' coin:

$$A = \left(g_1^{id_U} g_2\right)^s$$

$$B = g_1^{x_1} g_2^{x_2}$$

$$A, B, sig(A, B)$$

Showing coin:

$$r_1 = d(id_U s) + x_1$$

$$r_2 = d * s + x_2$$

Double spending:

$$r'_1 = d'(id_U s) + x_1$$

$$r'_2 = d' * s + x_2$$

$$id_U = \frac{r_1 - r'_1}{r_2 - r'_2} = \frac{(d - d')id_U s}{(d - d')s}$$

Brands-Based TAT System

Brands' coin:

$$A=\left(g_1^{id_U} g_2\right)^s$$
$$B=g_1^{x_1} g_2^{x_2}$$

$$A, B, sig(A, B)$$

Showing coin:

$$r_1=d\left(id_U s\right)+x_1$$
$$r_2=d*s+x_2$$

Double spending:

$$r'_1=d'\left(id_U s\right)+x_1$$
$$r'_2=d'*s+x_2$$

P4R' coin:

$$A=\left(g_1^{id_U} g_2\right)^s$$
$$B=g_1^{x_1} g_2^{x_2}$$
$$C=g_1^{x'_1} g_2^{x'_2}$$
$$A, B, C, sig(A, B, C)$$

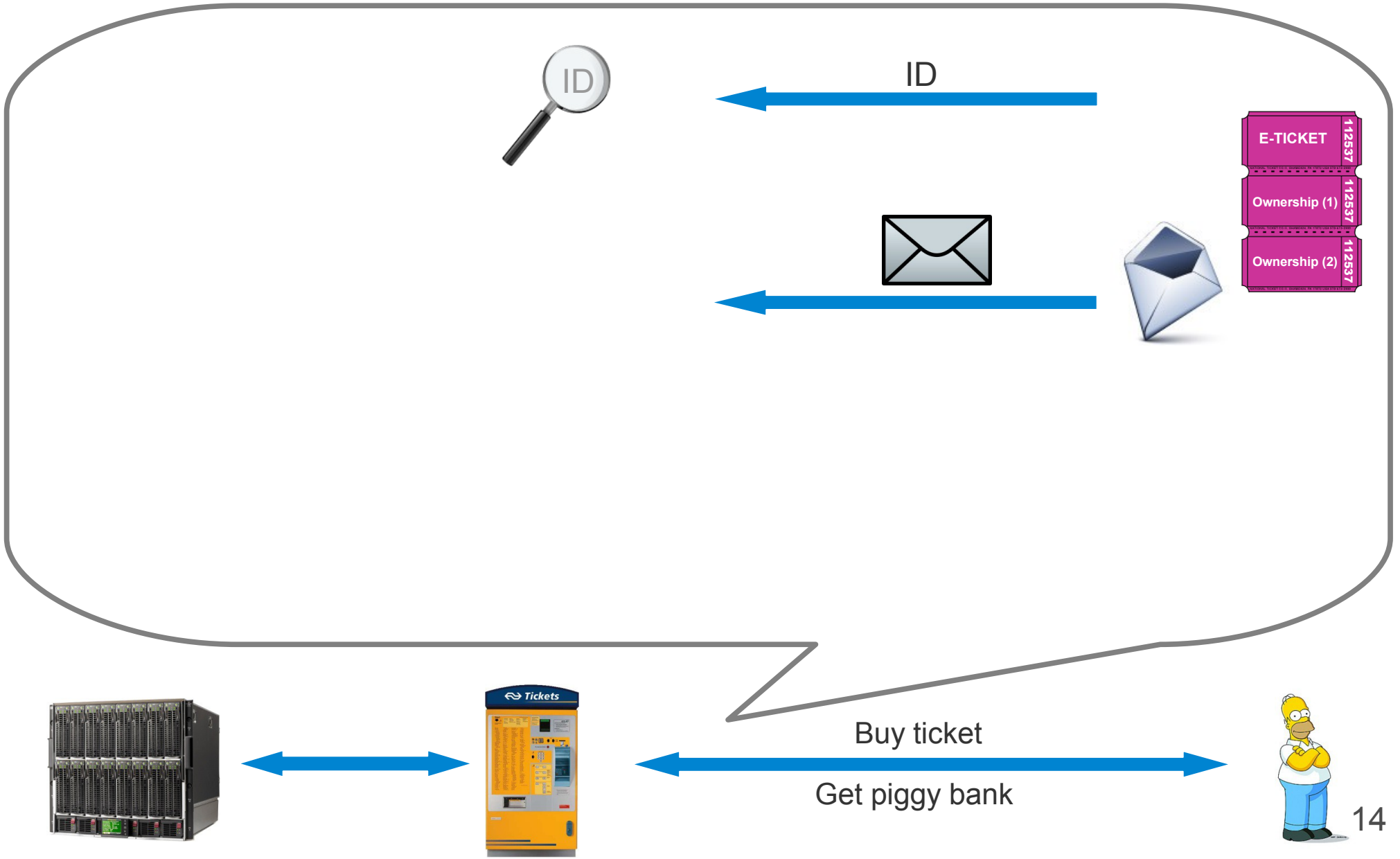
First spending:

$$r_1=d\left(id_U s\right)+x_1$$
$$r_2=d*s+x_2$$

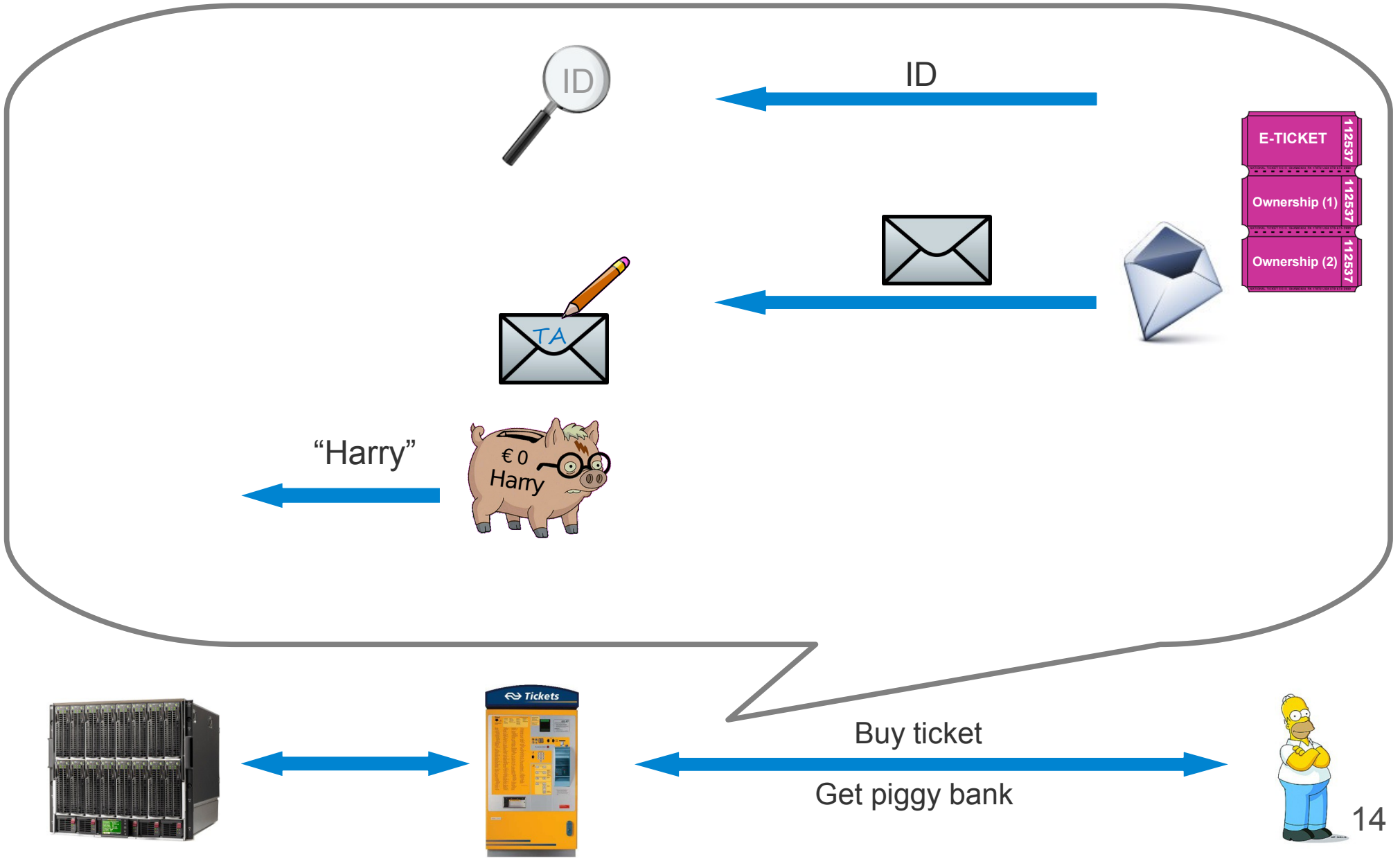
Second spending:

$$r'_1=d'\left(id_U s\right)+x'_1$$
$$r'_2=d'*s+x'_2$$

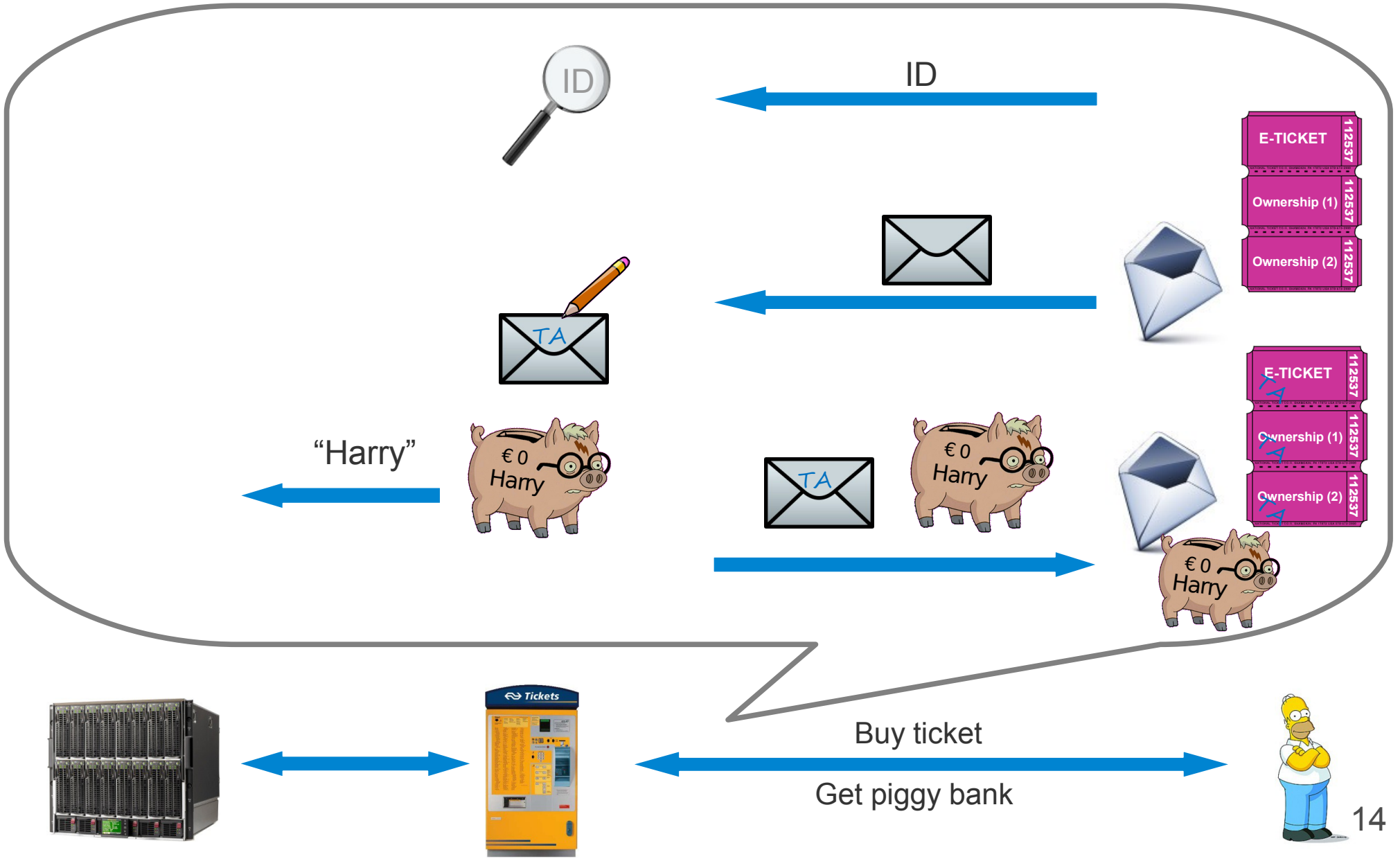
P4R: BuyTAT and GetRT



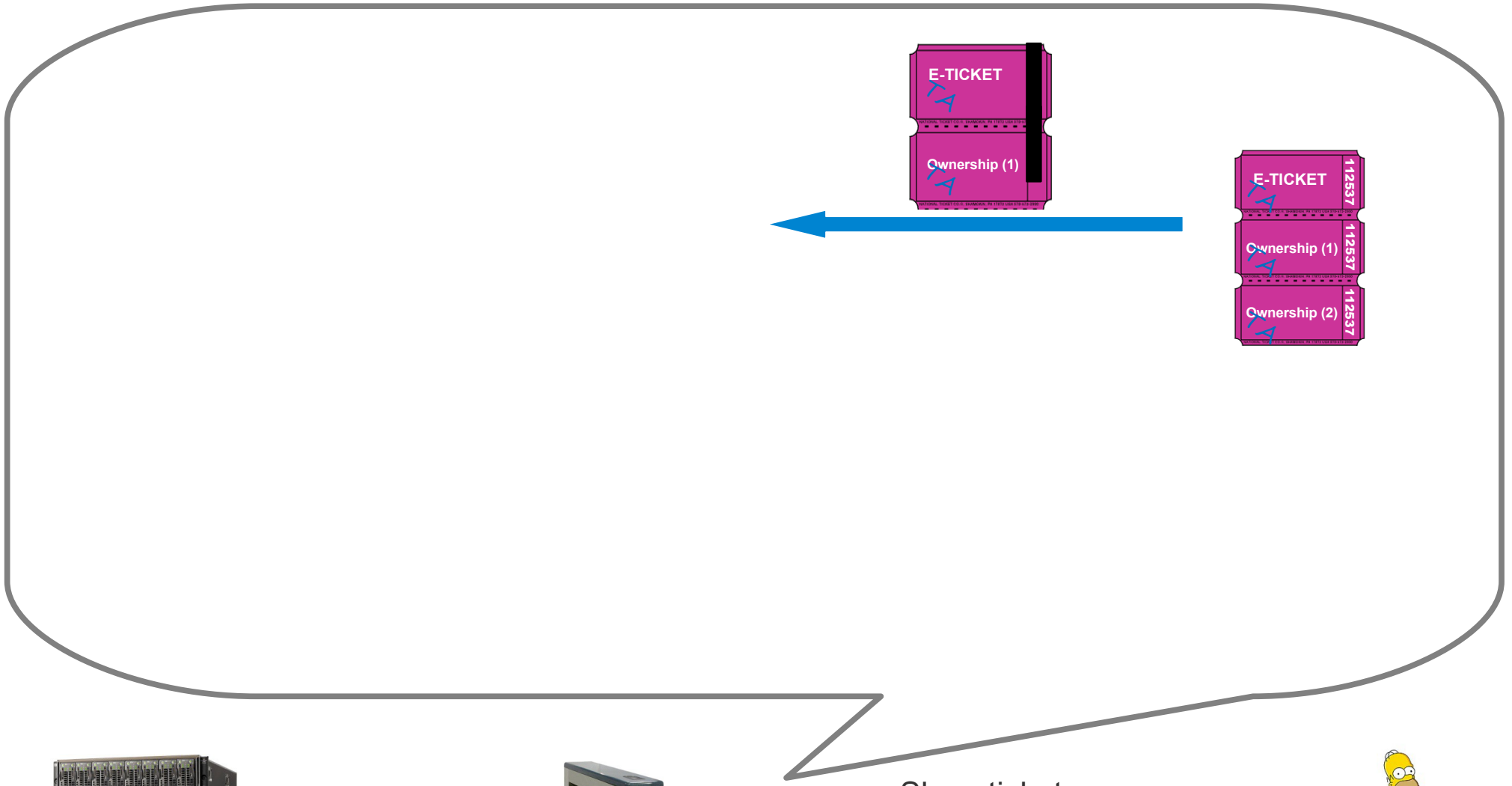
P4R: BuyTAT and GetRT



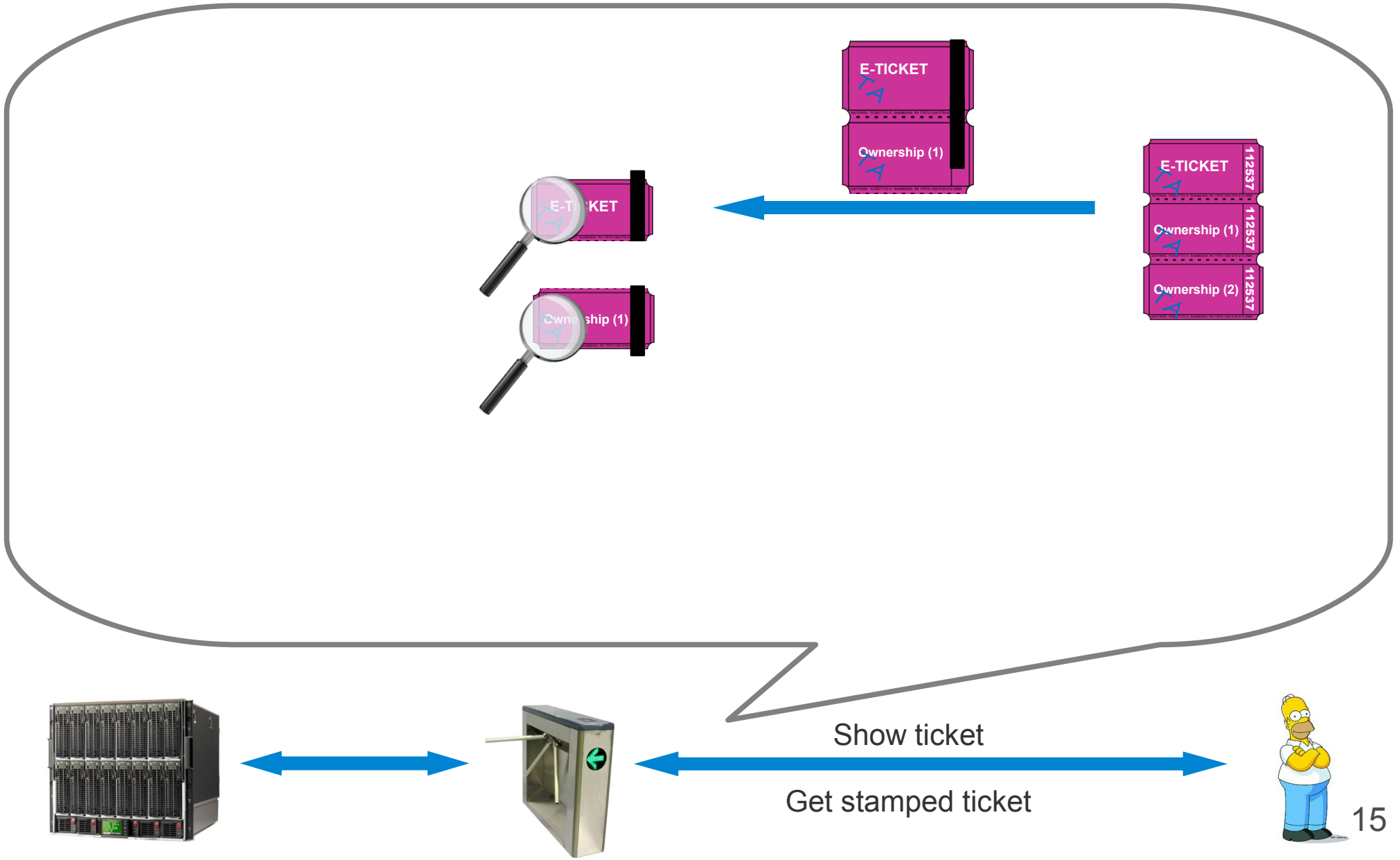
P4R: BuyTAT and GetRT



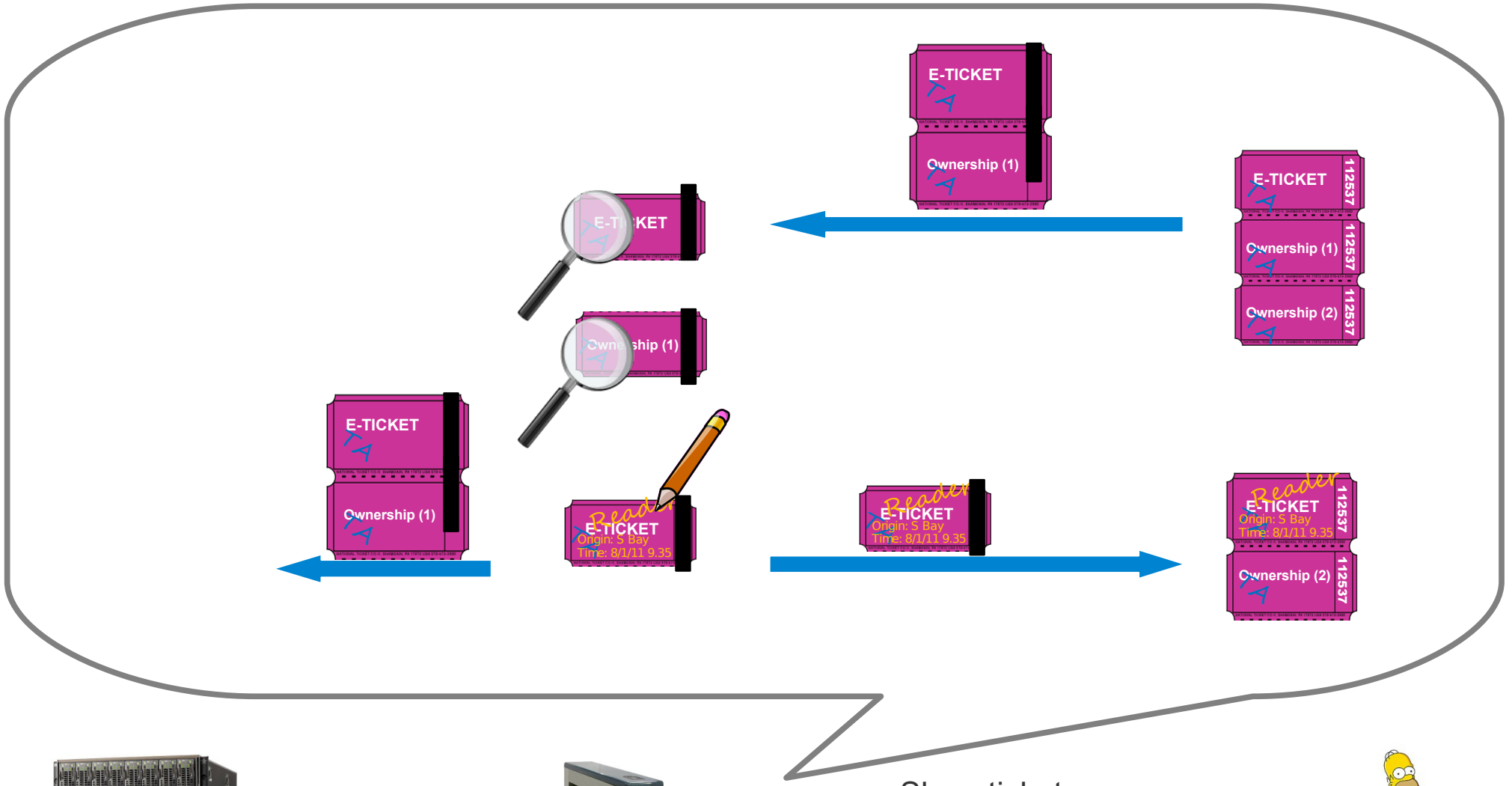
P4R: ShowTAT and GetRCT



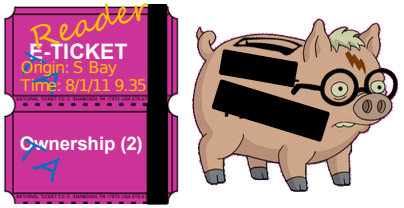
P4R: ShowTAT and GetRCT

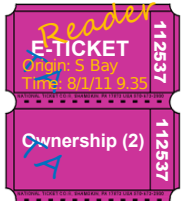


P4R: ShowTAT and GetRCT



P4R: ShowRCT and GetRefund

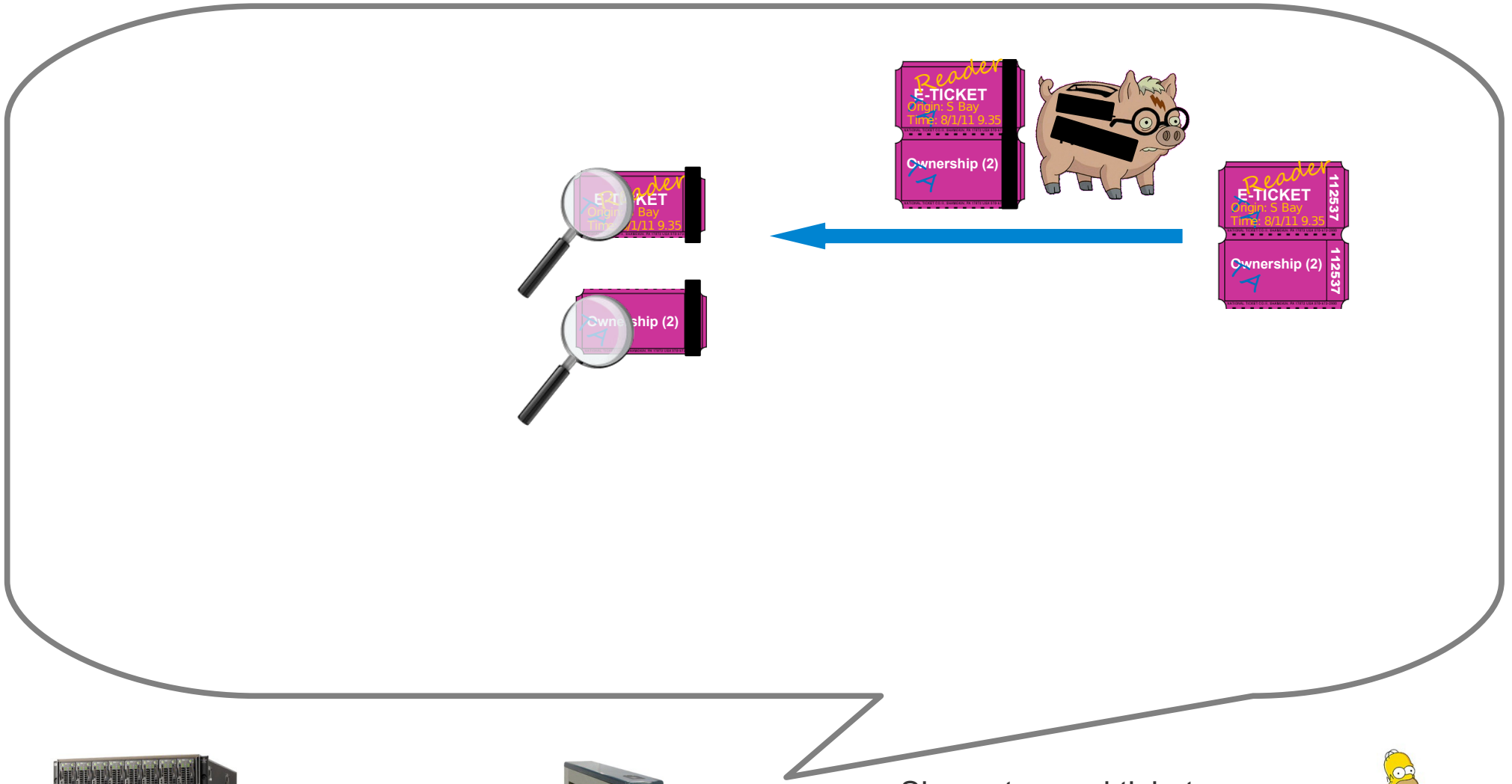




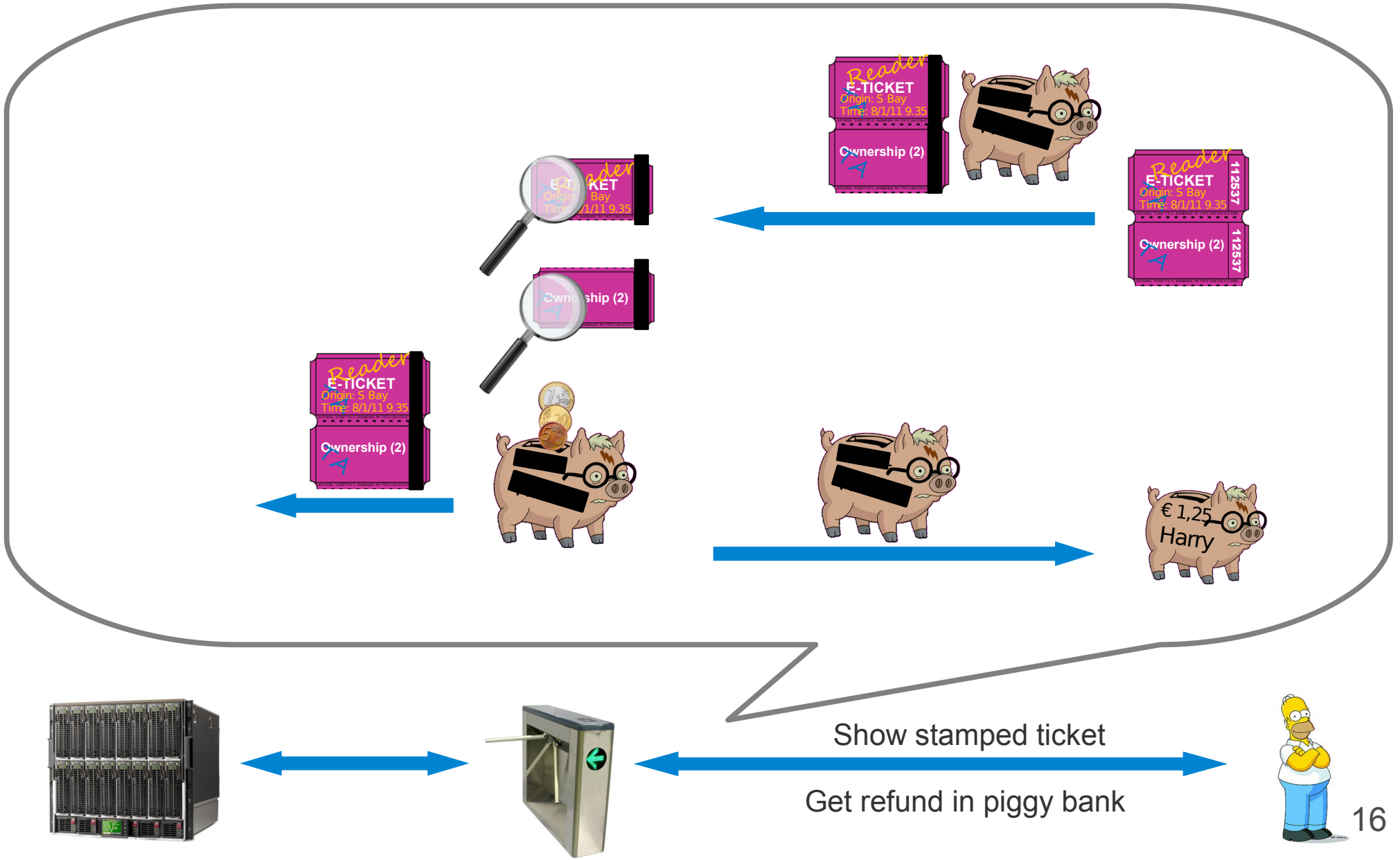
Show stamped ticket

Get refund in piggy bank

P4R: ShowRCT and GetRefund



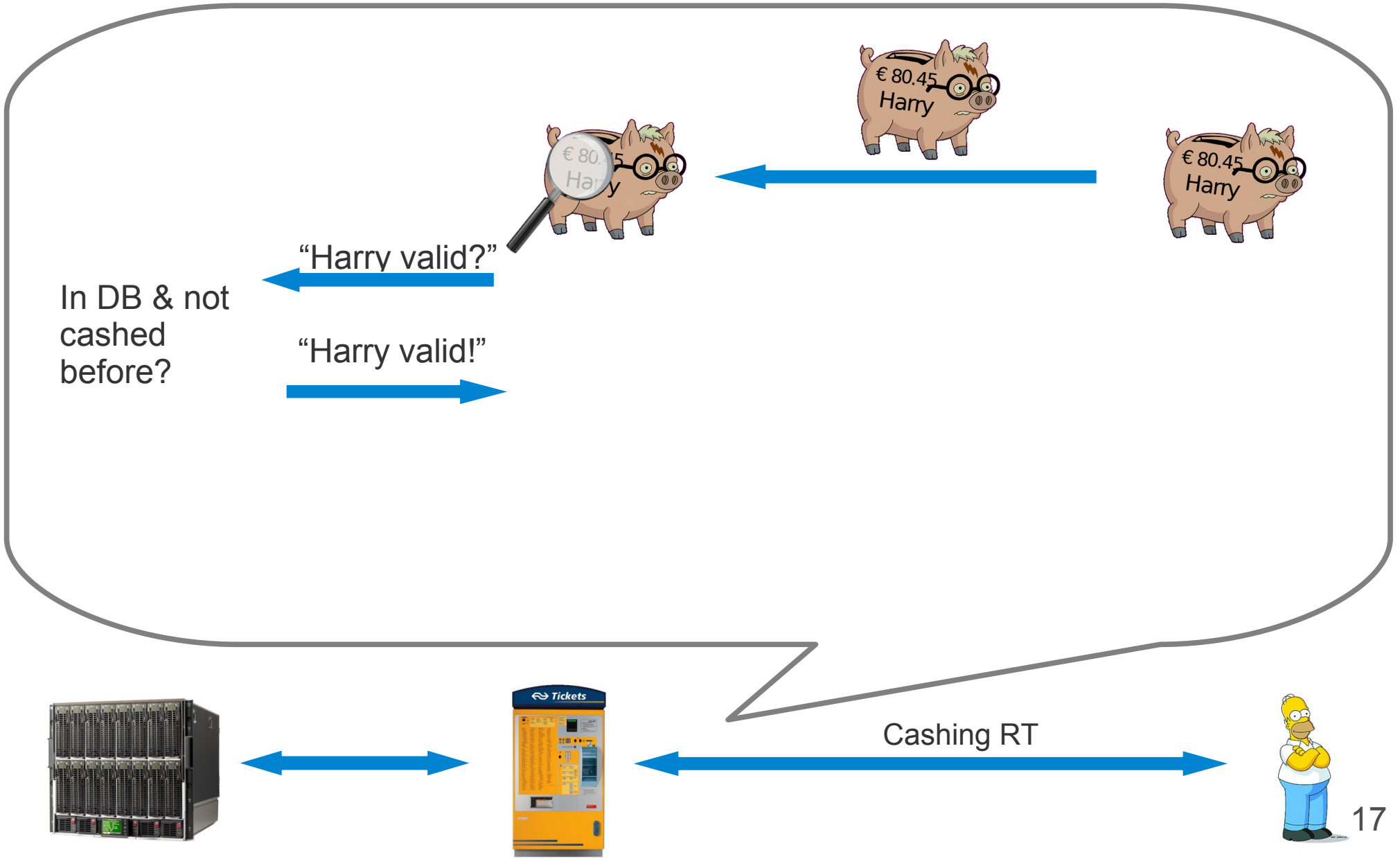
P4R: ShowRCT and GetRefund



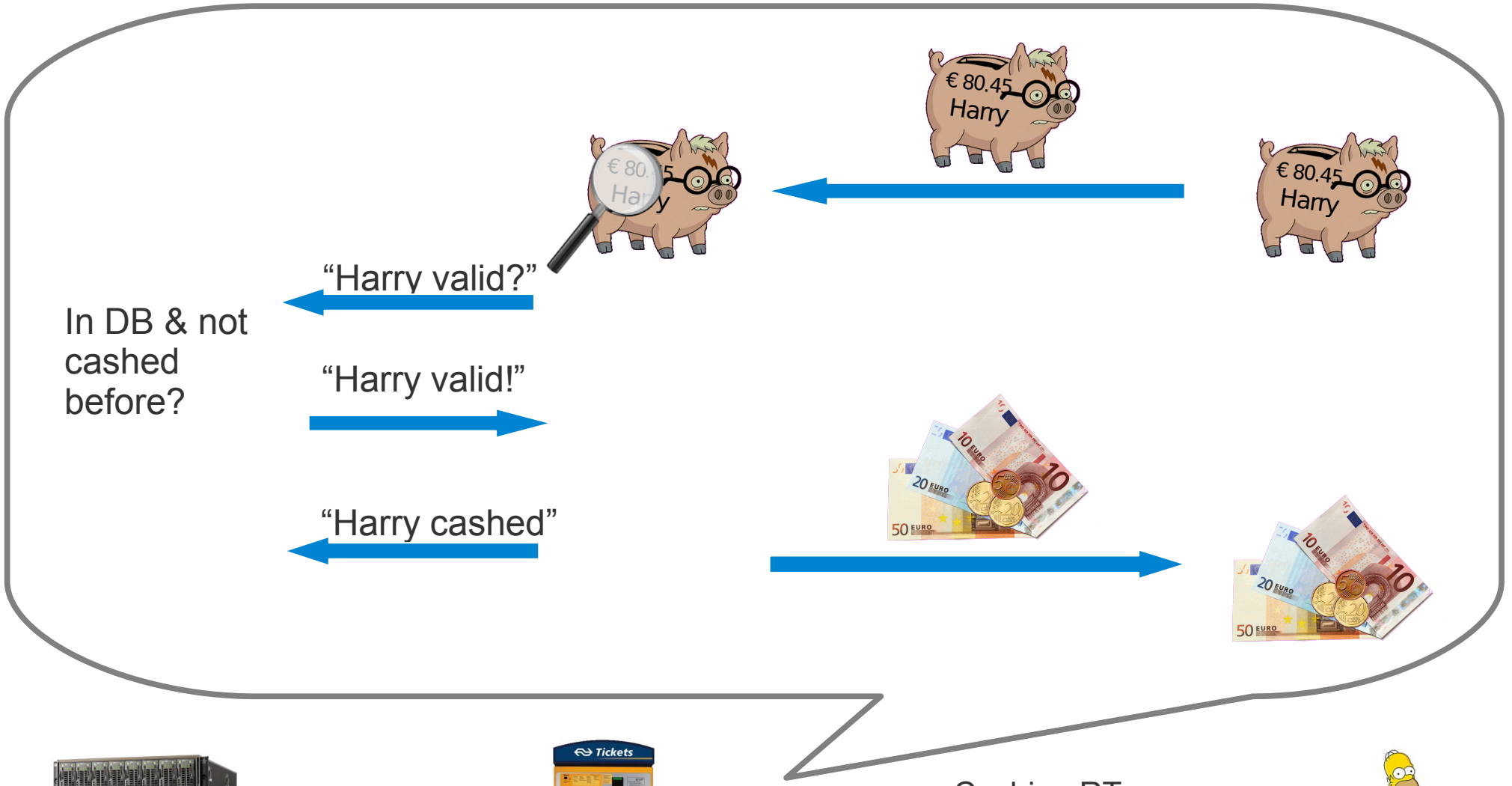
P4R: RedeemRT

Cashing RT

P4R: RedeemRT



P4R: RedeemRT



BLS-Signature Based RT System

A pairing is a bilinear map:

$$e(a^u, b^v) = e(a, b)^{uv} \quad \text{for all} \quad u, v, \in \mathbb{Z}_p, a, b, \in G_p$$

BLS-signatures requires an efficiently computable, non-degenerate pairing!

Boneh-Lynn-Shacham Signatures:

Keys: $sk = x \in \mathbb{Z}_p, v = g^x$

Signature on $m \in G$: $\sigma := H(m)^x$

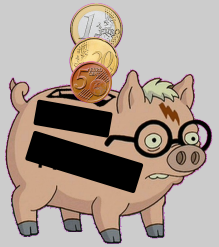
Verification of (m, σ): $e(g, \sigma) \stackrel{?}{=} e(v, H(m))$

BLS-Signature Based RT System

Refund token: $RT = Harry \in G,\ R = 1,\ v = 0$

Adding refund w user: $r \in \mathbb{Z}_p,\ RT' = RT^r,$
$v = v + w,\ R = R * r \bmod p$

Adding refund w TA: $RT' = RT'^{dw}$



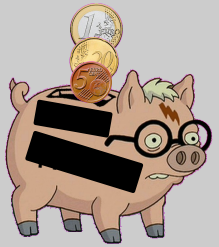
Verify claim for refund v: $e\left(Harry^R, h^{dv}\right) \stackrel{?}{=} e\left(RT', h\right)$

BLS-Signature Based RT System

Refund token: $RT = Harry \in G,\ R = 1,\ v = 0$

Adding refund w user: $r \in \mathbb{Z}_p,\ RT' = RT^r,$
$v = v + w,\ R = R * r \bmod p$

Adding refund w TA: $RT' = RT'^{d^{\sum w_i}}$



Verify claim for refund v: $e\left(Harry^R, h^{d^v}\right) \stackrel{?}{=} e\left(RT', h\right)$

Security of P4R

TA Security: TA does not lose any money

- User cannot forge tickets
- User cannot receive reimbursement that exceeds the overall deposit for tickets minus overall fare of trips

User Security:

- A passive adversary cannot steal tickets or refunds from a user

User Privacy:

- Adversary cannot differentiate between all possible trip sequences leading to the same total refund amount

User's Side Implementation on Moo

Storage space to make 20 trips is at most 7.62 KB!

	Cycle count	Execution time @16 MHz in s
BuyTAT & GetRT	84,585,590	5.29
ShowTAT & GetRCT	35,264	0.002
ShoeRCT & GetRefund	5,466,485	0.34
RedeemRT*	5,549,538	0.35

* Excludes authenticating to the vending machine.

Thank you for your attention!!!