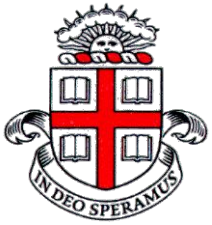
A Whirlwind Tour of Anonymous Credentials and Related Protocols

Anna Lysyanskaya
Brown University



Anonymous Credentials: Motivation

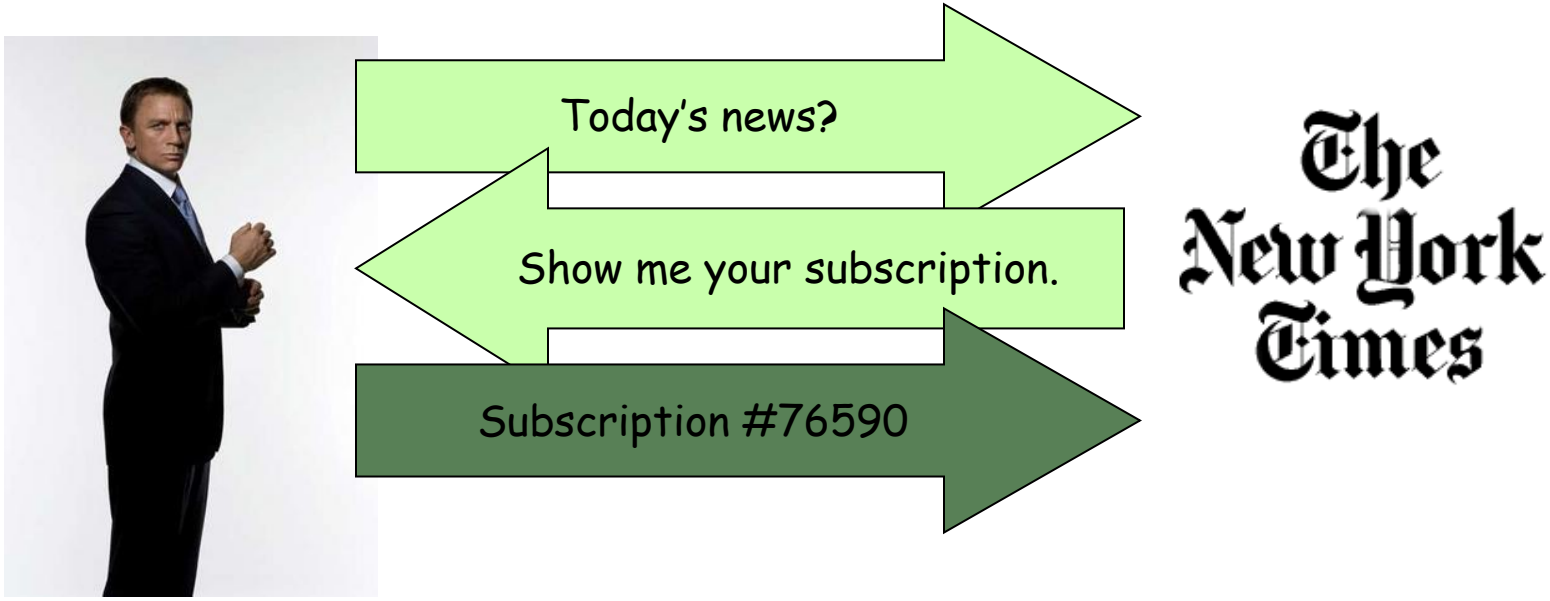
Today's news?

Who are you? Do you have a subscription?

I can tell you, but then I'll have to kill you...

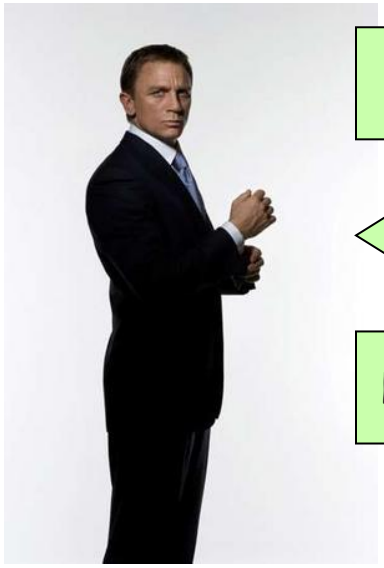
The New York Times

Anonymous Credentials: Motivation



87% of US population uniquely identifiable by date of birth, zip code and gender [Sweeney].

Anonymous Credentials: Motivation



Today's news?

Prove that you are authorized.

Here is a zero-knowledge proof

The New York Times

Anonymous Credentials: Definition of Security

Just kidding! Instead, I'll show you how to construct them so as to satisfy any reasonable definition...

(Don't generally recommend this approach.)

What's Under the Hood?

Credential issue:

I am PK_{JB}. Please give me a cert that says I have a subscription

$\sigma_{NYT}=\sigma_{NYT}(PK_{JB}, \text{Subscription})$

The New York Times

PK_{NYT}

What's Under the Hood?

Credential issue:

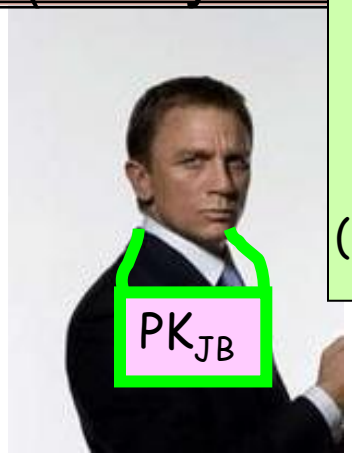
I am PK_{JB}. Please give me a cert that says I have a subscription

$\sigma_{NYT}=\sigma_{NYT}(PK_{JB}, \text{Subscription})$

The New York Times

PK_{NYT}

Credential demo (anonymous):



Zero-knowledge proof that

I know SK, PK and σ such that:

(1) PK corresponds to SK

(2) Verify(PK_{NYT},(PK. Subscription),σ).

The New York Times

What's Under the Hood?

Credential issue (anonymous):

PK_{ProJo}

SK_{JB}, creds

$\sigma = \sigma_{ProJo}(SK_{JB})$

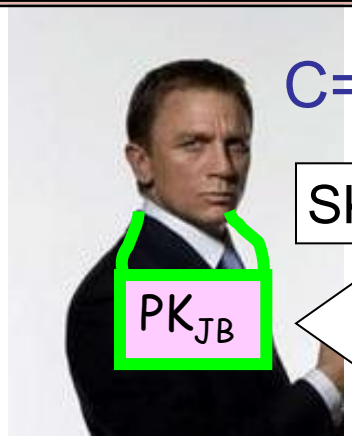
2PC

SK_{ProJo}

The Providence Journal

What's Under the Hood?

Credential issue (anonymous):



$C=Commit(SK_{JB},R)$

C, PK_{ProJo}

SK_{JB}, R, creds

$\sigma = \sigma_{ProJo}(SK_{JB})$

2PC

SK_{ProJo}

The Providence Journal

PK_{ProJo}

PK_{JB}

In theory, we are done...

- Anonymous issuing:
  - Bond's pseudonym is C = Commit(SK_{JB}; R)
  - Credential $\sigma = \sigma_{ProJo}(C)$

- Anonymous demo:
  - [GMW+BG] ZK proof of knowledge for any NP relation
  - [DDP00+DDOPS01] "Robust" NIZK proof of knowledge for any NP relation
    - To a verifier who knows him by pseudonym C' = Commit(SK_{JB}; R'), Bond proves knowledge of (SK,C,R,R'σ) such that
      - C = Commit(SK; R) and C' = Commit(SK; R')
      - $\sigma = \sigma_{ProJo}(C)$

If we want this stuff used, then it’s another story...

What's Needed for Anonymous Credentials

- A commitment scheme and a signature scheme with three efficient protocols:
  - "Robust" ZK proof of knowledge and equality of committed values
  - "Secure" protocol for signing a committed value/a set of committed values
  - "Robust" ZK proof of knowledge of a signature on a set of committed values

History

- 1980s: Chaum's vision (no actual defs or constructions)
- 1990s: the naive era
  - Brands99: no proof of security, single-use
  - Damgard90,LRSW99: general inefficient constructions
- 2000s: the early modern era
  - CL01,L02,CL02,CL04,BBS04,BCKL08: identify the right building blocks and give efficient constructions under various complexity assumptions (strong RSA, LRSW, qSDH), from interactive to non-interactive in RO model and CRS models
  - CHL05,CHKLM06,BCKL09: ecash and etokens
- 2010s: the age of GS proofs
  - Gro06, AFGHO'10, AGHO'11, HJ12, ACDKNO'12: "structure-preserving signatures:" signatures on group elements that also consist of group elements, verification equations can be expressed as pairing product equations
  - BCCKLS09,CKLM13: delegatable anonymous credentials

History (the Practice, in $10Ms)

- IBM's Idemix project + European partners (2003-present):
  - outgrowth of [CL01]
  - funding from the EU, about 30M Euro so far
  - implementations, pilots

- Trusted Computing Group (TCG) standard (2004):
  - direct anonymous attestation (DAA) uses my anonymous credentials – hardware support on every PC

- Microsoft's UProve (2007-present):
  - bought Stefan Brands' company for undisclosed amount of money

- National Strategy for Trusted Identities in Cyberspace (NSTIC) (2011):
  - comes from the White House
  - administered by NIST, about $20M

Pandora's Box

- Anonymity is an invitation for abuse. Alice will share her credentials with all of her friends.
  - Answer #1: anonymity is not the issue. The fact that credentials are digital is the issue.
  - Answer #2: can have limited-use credentials.
  - Answer #3: can revoke credentials in case of abuse, similarly to non-anonymous case.
  - Answer #4: can escrow Alice's identity, to be revealed in case of emergency.
  - Answer #5: can make Alice's SK too valuable to share.

Roadmap

- Warm-up: commitment, signature, protocols from CL04+BBS04
- Structure-preserving signature (SPS)
- “Robust” NIZK PoK from GS NIWI and SPS [adapted from Groth06]
- Anonymous credentials from SPS and “robust” NIZK

---- break ----

- Ecash from these building blocks [adapted from BCKL09]
- Delegatable anonymous credentials [BCCKLS09,CKLM13]

q-SDH Assumption in BM Groups

- Given: G, G_T of order q, g of G, BM e: $G \times G \rightarrow G_T$, values $\{X_i = g^{x\hat{}i}: 1 \leq i \leq q\}$
- Hard to compute (a, A) such that $A = g^{1/(x+a)}$.

- [BBS04]: The following sig scheme is secure against non-adaptive attack under q-SDH:
  - key generation: PK = (G, G_T, e, g, X), SK = $(x : X = g^x)$
  - signature on a is $g^{1/(x+a)}$.
  - verification of (a,A): $e(Xg^a, A) = e(g,g)$.

- (non-adaptive attack means that an adversary sees up to q signatures on random a's)

CMA-Secure Sig for Blocks

- Non-adaptive sig:
  - key generation: PK = (G, G_T, e, g, X), SK = $(x : X = g^x)$
  - signature on a is $A = g^{1/(x+a)}$. ie $A^{x+a} = g$
  - verification of (a, A): $e(Xg^a, A) = e(g,g)$.

- Modification [BBS04 + CL04]:
  - keygen: PK = $(G, G_T, e, g, g_0, Z_1, \ldots, Z_L, X)$,
    SK = $(x : X = g^x)$
  - signature on $(r, m_1, \ldots, m_L)$ is (A, a) such that
    $A^{x+a} g_0^r \prod Z_i^{mi} = g$
    (signer picks random a and solves for A to compute sig)
    NOTE: to sign, sufficient to know $M = g_0^r \prod Z_i^{mi}$
  - verification: $e(Xg^a, A) = e(g, g/ g_0^r \prod Z_i^{mi}) = e(g, g/M)$.

CMA-Secure Sig for Blocks

- Non-adaptive sig:
  - K
  - g
  - V

g^x)

In the proof of security, the reduction is given a non-adaptive sig (b,B) and all the dlogs: $g_0 = g^u$ and $Z_i = g^{v_i}$ and it must solve for a s.t.: $a + ru + \sum m_i v_i = b$ and output (a,B) as the sig

- Modification [BBS04 + CL04]:
  - keygen: PK = $(G, G_T, e, g, g_0, Z_1, \ldots, Z_L, X)$,
    SK = $(x : X = g^x)$
  - signature on $(r, m_1, \ldots, m_L)$ is (A, a) such that
    $A^{x+a} g_0^r \prod Z_i^{m_i} = g$
    (signer picks random a and solves for A to compute sig)
    NOTE: signer need not know $(r, m_1, \ldots, m_L)$, only $g_0^r \prod Z_i^{m_i}$
  - verification: $e(Xg^a, A) = e(g, g / g_0^r \prod Z_i^{m_i})$.

And the Protocols:

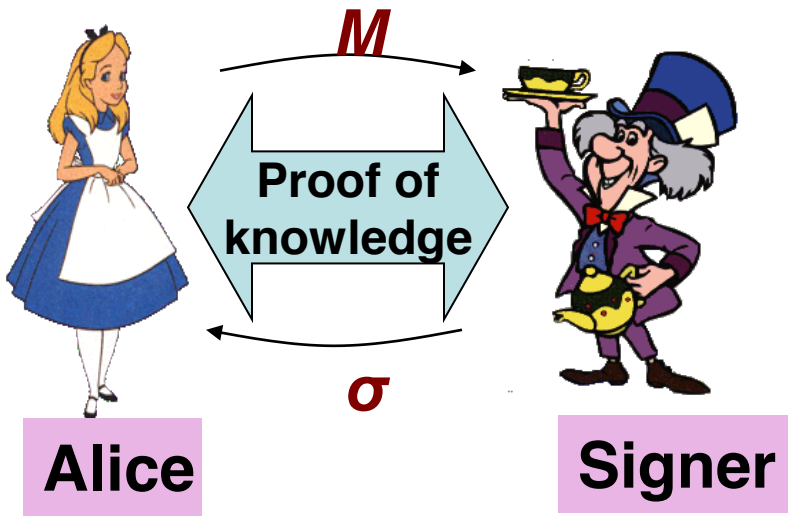
(1) obtaining sig on a committed value
(2) ZKPOK of a sig on a committed value
(use Pedersen commitments)

Signature on a Committed Value

1. Commit to m:

   $M = g_0^r \prod Z_i^{mi}$

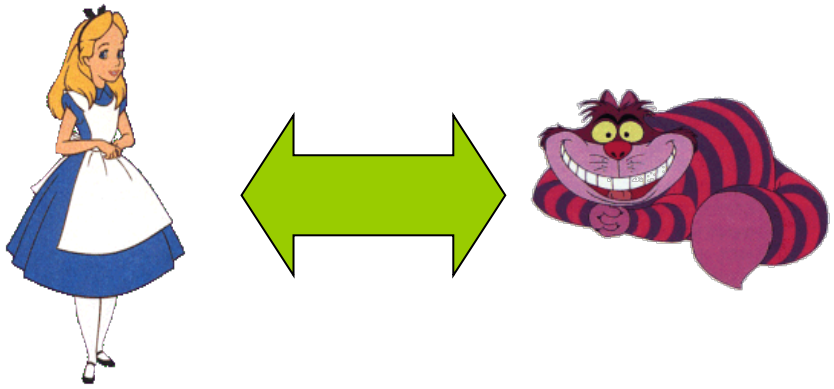
2. ZKPOK of representation of M in g_0, Z



3. Issue the signature σ

4. Output signature!

Proof of Knowledge of a Signature



- Idea: Prover holds $a, r, \{m_i\}, A$ such that
  $e(Xg^a, A) = e(g, g/M)$
  $M = g_0^r \Pi Z_i^{mi}$
- Express everything as a relationship between discrete logarithm representations & use [Schnorr91,Brands99] (interactive or RO model)

Why don't want interactive proofs?

- Just don't. Interaction is expensive.
- Composition issues: complicated to get knowledge extraction without rewinding, making it work makes it much more expensive [CS03]; standard constructions for UC-secure ZK are based on "robust" NIZK [CF01,CLOS02].
- Interactive proofs are non-transferable, don't work for some applications (e.g. ecash, delegatable credentials).

Structure-Preserving Signatures

- First appeared in [Gro06]. Better constructions are [AFGHO'10,AGHO'11,HJ12,ACDKNO'12]. They are incomparable to each other: different assumptions and sizes. Most efficient has three group elements (necessary) [AGHO'11].

- Definition: a secure signature scheme (Paramgen, Keygen,Sign,Verify) is structure-preserving if:
  - PKs, messages, and sigs are sets of elements of G_1 or G_2 for which there's a bilinear map e: G_1 x G_2 -> G_T
  - Verify checks a pairing prod equation (PPE) of the form

    $\Pi_i \Pi_j e(A_i,B_j)^{aij} = 1$, where $\{A_i\}$ in G_1, $\{B_j\}$ in G_2, are elements of parameters, PK, message or signature, and a_{ij} are integer constants

Application of SPS [Gro06]: Simulation-Extractable NIZK from GS NIWI (1 of 2)

- "Definition" [SimExt NIZK PoK]: NIZK PoK where adversary A can't win this game:
  - A adaptively requests simulated proofs of statements of his choice
  - A outputs (new statement x, proof π)
  - KnowledgeExtractor(x,π) computes w
  - A wins if w is NOT a witness for x
- Stronger definition: A can't win even given the extraction trapdoor
- This is essentially the notion of "robust" NIZK we care about

Application of SPS [Gro06]: Simulation-Extractable NIZK from GS NIWI (2 of 2)

- Let PPE be a pairing product equation. Then

$$L_{PPE} = \{\{C_i\} \mid \text{values inside commitments } \{C_i\} \text{ satisfy the PPE}\}$$

  - Recall: GS NIWI designed for languages of this form, for extractable commitments; has perfect soundness/extractability
- SimExt NIZK for L_{PPE}:
  - CRS contains GS NIWI CRS_1, and a PK for a SPS
  - Prover forms new commitments $\{C'_i\}$ and uses GS NIWI to prove that either $\{C_i\}$ in L_{PPE} or values inside $\{C'_i\}$ are a signature under PK on the values $\{C_i\}$ Exercise: express disjunctions as PPEs
  - Simulator has SK for PK, forms proofs by signing $\{C_i\}$
  - If A outputs a new statement x, and extractor can't extract the witness attesting that x in L_{PPE}, then by perfect extraction properties of GS NIWI it extracts a new signature – contradiction! (Works even if A knows the extraction trapdoor.)

Anonymous Credentials from SPS and SimExt NIZK PoK [BCKL08,...,CKLM13]

- System parameters: CRS for SimExt NIZK PoK for PPEs (what we just saw)
- Issuer's PK: PK for a structure-preserving sig

Credential issue (anonymous):

$\sigma = \sigma_{PK}(Nym)$

The Providence Journal

PK_{ProJo}

- where $Nym = Commit_{GS}(SK_{JB})$
  - Nym and σ consist of group elements.

Anonymous Credentials from SPS and SimExt NIZK PoK [BCKL08,...,CKLM13]

Credential demo (anonymous):

C_{Nym} = $Commit_{GS}$(Nym)
C_{σ}= $Commit_{GS}$(σ)
Proof π that (1) the value inside C_{σ} is a sig on the value inside C_{Nym} (2) the value inside C_{Nym} is a commitment to the same value as the one inside Nym'

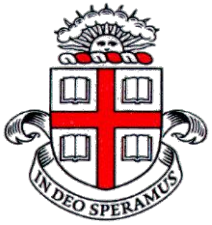
The New York Times

Anonymous Credentials from SPS and SimExt NIZK PoK [BCKL08,...,CKLM13]

- System parameters: CRS for SimExt NIZK PoK for PPEs (what we just saw)
- Issuer's PK: PK for a structure-preserving sig
- Issue: Let Nym = $\text{Commit}_{GS}(SK_{JB})$. Issuer computes $\sigma = \sigma_{PK}(\text{Nym})$ and sends it to James Bond.
  - Recall: Nym and σ consist of group elements.
- Demo: Let Nym' = $\text{Commit}_{GS}(SK_{JB})$ (another pseudonym for James Bond). James Bond wants to prove that the identity inside Nym' has a credential from the Issuer. His proof π consists of the following:
  - $C_{\text{Nym}} = \text{Commit}_{GS}(\text{Nym})$, $C_\sigma = \text{Commit}_{GS}(\sigma)$
  - Proof π that (1) the value inside C_σ is a sig on the value inside C_{Nym} and (2) the value inside C_{Nym} is a commitment to the same value as the one inside Nym'.

A Whirlwind Tour of Anonymous Credentials and Related Protocols, Part 2

Anna Lysyanskaya
Brown University



Roadmap

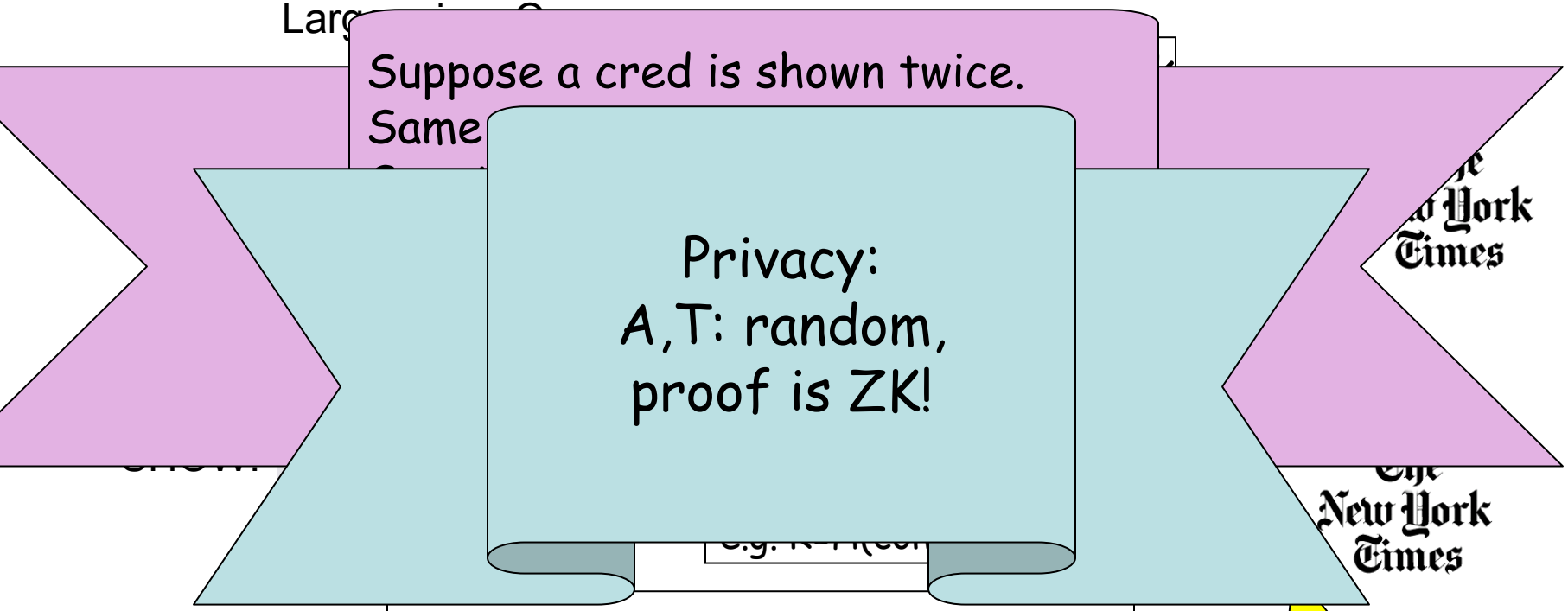
- Warm-up: commitment, signature, protocols from CL04+BBS04
- Structure-preserving signature (SPS)
- “Robust” NIZK PoK from GS NIWI and SPS [adapted from Groth06]
- Anonymous credentials from SPS and “robust” NIZK

---- break ----

- Ecash from these building blocks [adapted from BCKL09]
- Delegatable anonymous credentials [BCCKLS09,CKLM13]

Single-Use Creds (Idea) [CFN88,Brands]

- SETUP: Signature key pair for Issuer (pk,sk).
  Assume a PKI for all the users.
  Larg



A (credential's serial number)
T =x+RB mod Q (double-spending equation)

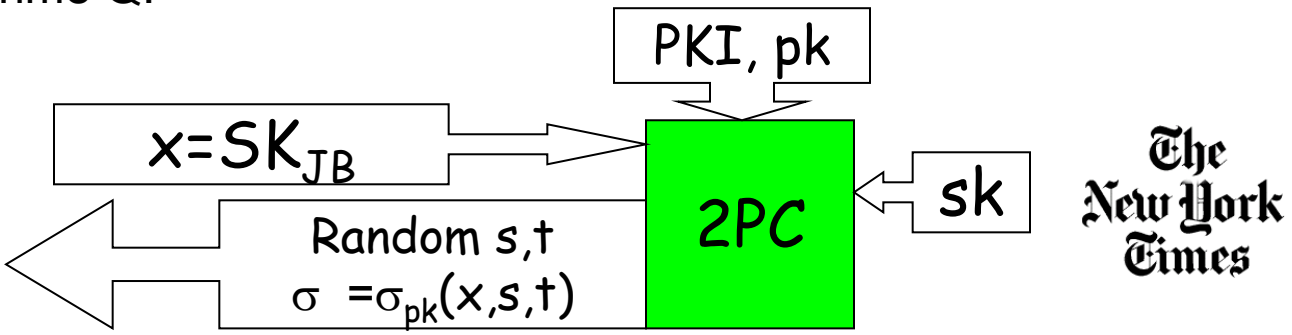
ZK proof of knowledge of (x,B,σ) such
1. T = x+RB
2. VerifySig(pk,(x,A,B), σ) = TRUE

Store
(A,R,T,proof)

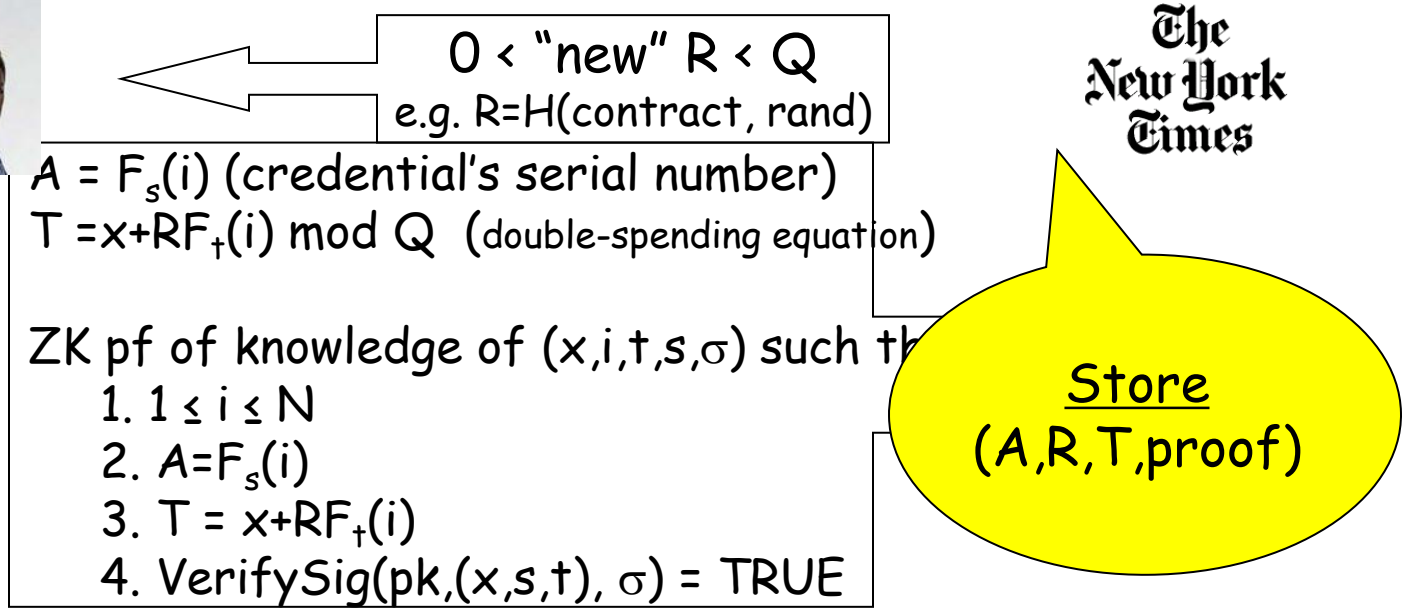
N-Use Creds/Compact Ecash [CHL05]

- SETUP: Signature key pair for Issuer (pk,sk).
  Assume a PKI for all the users.
  Large prime Q.

- ISSUE:



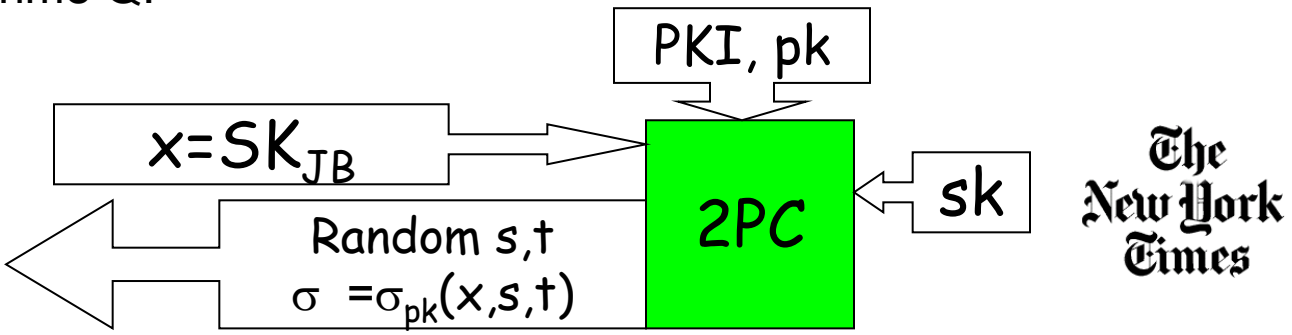
- SHOW i^{th} time:



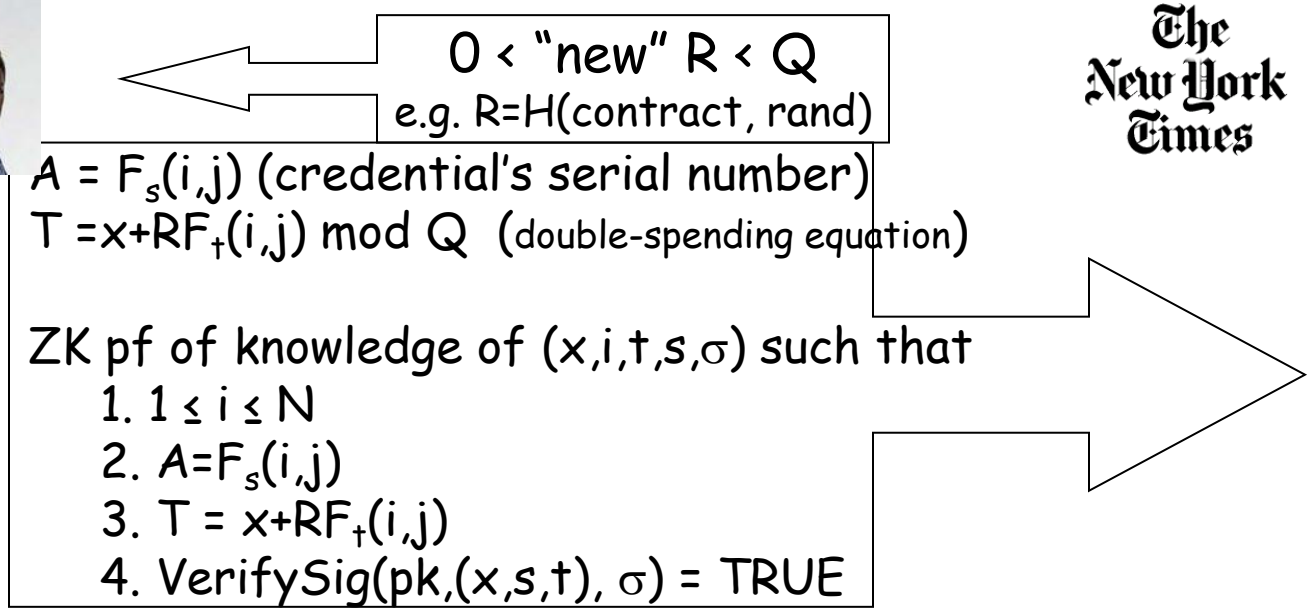
N per Day Creds/Anon. Etokens [CHKLM06]

- SETUP: Signature key pair for Issuer (pk,sk).
  Assume a PKI for all the users.
  Large prime Q.

- ISSUE:



- SHOW i^{th} time on Day j:



Pandora's Box

- Anonymity is an invitation for abuse. Alice will share her credentials with all of her friends.
  - Answer #1: anonymity is not the issue. The fact that credentials are digital is the issue.
  - Answer #2: can have limited-use credentials.
  - Answer #3: can revoke credentials in case of abuse, similarly to non-anonymous case.
  - Answer #4: can escrow Alice's identity, to be revealed in case of emergency.
  - Answer #5: can make Alice's SK too valuable to share.

Before GS Proofs

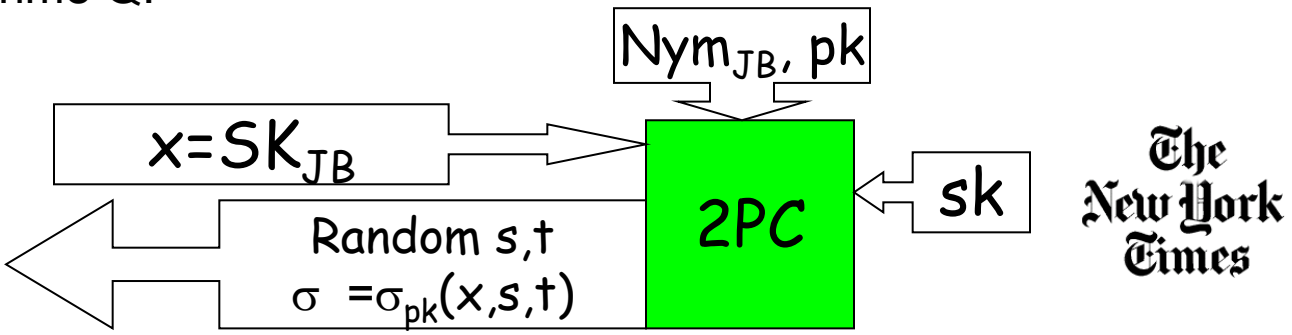
- In theory: could instantiate using general robust NIZK, get provably security
  - inefficient, useless for practical applications ☹

- In practice:
  - could instantiate under various number-thretic assumptions
  - use the Fiat-Shamir transform to get a NIZK
  - sacrifice provable security ☹

- With GS proofs:
  - the best of both worlds ☺

How to Instantiate Compact Ecash?

- SETUP: Signature key pair for Issuer (pk,sk).
  Assume a PKI for all the users.
  Large prime Q.
- ISSUE:



- [BCKL09]: GS-proof-based instantiation without SPS's
- Adapted from [BCKL09], but with an SPS (easier):
  - Step 1: They agree on commitments C_s, C_t to random s and t using coin-flipping; Bond knows openings s and t
    NB1: s and t are integers, not group elements!
    Open question: "structure-preserving" PRF
    NB2: AFAIK this requires interaction with the Issuer
  - Step 2: Bond obtains $\sigma = \sigma_{pk}(Nym_{JB},C_s,C_t)$

How to Instantiate (continued)?

- SHOW i^{th} time:

0 < "new" R < Q
e.g. R=H(contract, rand)

The New York Times

$A = F_s(i)$ (credential's serial number)
$T = x + RF_t(i) \bmod Q$ (double-spending equation)

ZK pf of knowledge of (x,i,t,s,σ) such that
1. $1 \le i \le N$
2. $A = F_s(i)$
3. $T = x + RF_t(i)$
4. VerifySig(pk,(x,s,t), σ) = TRUE

How to Instantiate (continued)?

- SHOW i^{th} time:

0 < "new" R < Q
e.g. R=H(contract, rand)

The New York Times

$A = F_s(i)$ (the coin's serial number)
$T = x+RF_t(i) \bmod Q$ (double-spending equation)

SimExt NIZK PoK of $(C_i, x, C_x, C_s, C_t, \sigma)$ such that
- ☑ 0. C_i, C_x, C_s, C_t are commitments to i,x,s,t
- 1. $1 \le i \le N$
- 2. $A = F_s(i)$
- 3. $T = x+RF_t(i)$
- ☑ 4. VerifySig(pk,(C_x, C_s, C_t), σ) = TRUE

How to Instantiate (continued)?

- SHOW i^{th} time:

0 < "new" R < Q
e.g. R=H(contract, rand)

The New York Times

$A = F_s(i)$ (the coin's serial number)
$T = g^x F_t(i)^R$ (double-spending equation)

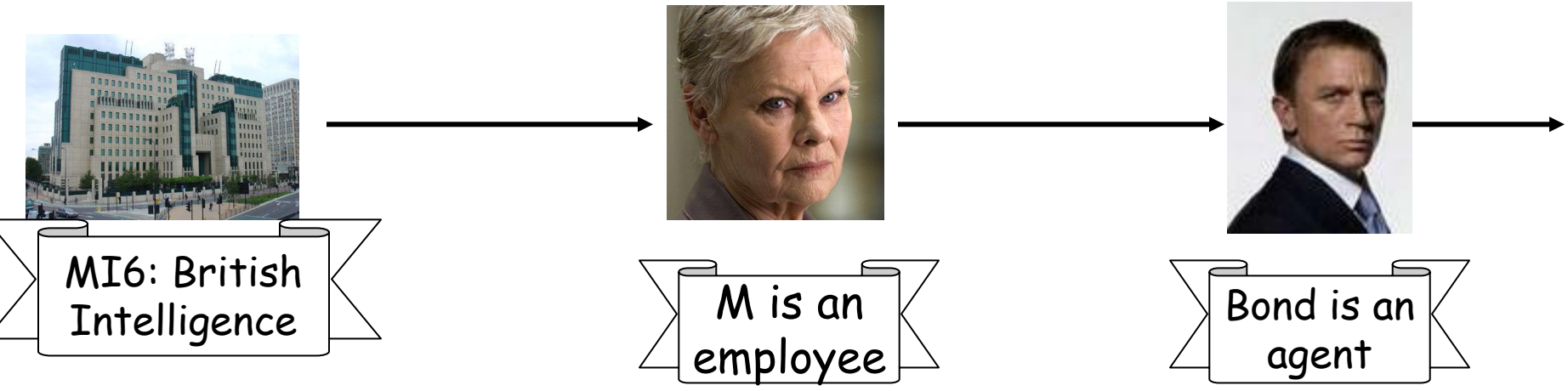
SimExt NIZK PoK of $(C_i, x, C_x, C_s, C_t, \sigma)$ such that
- ☑ 0. C_i, C_x, C_s, C_t are commitments to i,x,s,t
- ☑ 1. $1 \le i \le N$ various standard techniques
- ☑ 2. $A = F_s(i)$
- ☑ 3. $T = g^x F_t(i)^R$
- ☑ 4. VerifySig(pk,(C_x, C_s, C_t), σ) = TRUE

Use DY05 PRF: $F_s(i) = g^{1/(s+i)}$, can express correctness of A and T as PPEs

But…

- These credentials are a simplification of what non-anonymous credentials look like in practice!

Credential Chains



- Non-anonymous: trivial from signatures + ID schemes
- Bond's anonymous credential
  - Reveals that he got a credential from a valid employee (who had a credential from MI6)
  - Should reveal no other information
  - Even Bond himself should not know M's real name and her PK!
- (Compare with conventional certification chains…)

Credential Chains

MI6: British Intelligence

M is an employee

Bond is an agent

- M's credential: $\sigma = \sigma_{MI6}(PK_M, \text{M's attributes})$
- Bond's credential:
  - M must somehow use the fact that she knows her SK_M and the value σ to "sign" PK_{Bond}.

Efficient Anonymous Delegation via Randomizable NIZK proofs [BCCKLS09,CKLM13]

Randomizable Proofs [BCCKLS09]

- Randomizable commitment:

  Commit(x,r) $\mathcal{T}$ r' = Commit(x,r+r')

Efficiently computable operation

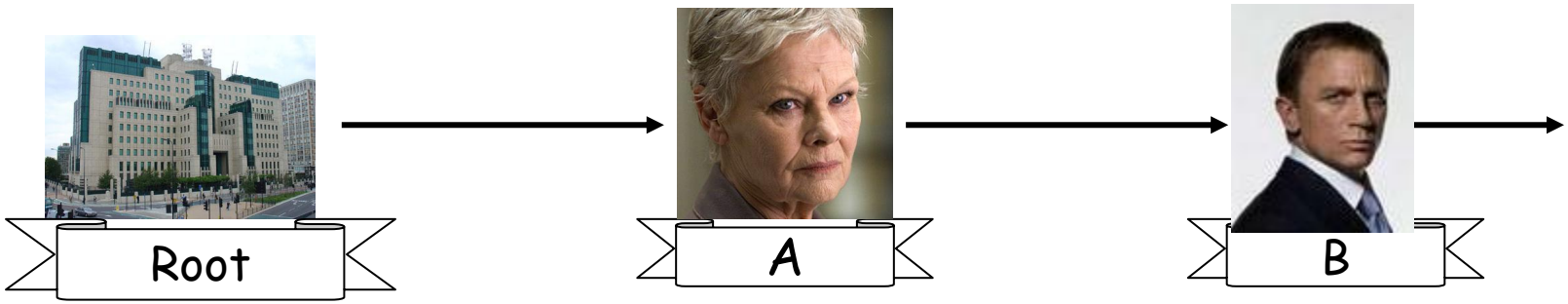
Random given one of r, r'

Randomizable Proofs [BCCKLS09]

- Randomizable commitment:
  Commit(x,r) $\mathcal{E}$ r' = Commit(x,r+r')
- NIZK proof system (Setup, Prove, Verify) that committed values satisfy relation R
- Algorithm Rand($C_1,\ldots C_n,\pi,r'_1,\ldots,r'_n$)-> π'

Randomizable if (1) and (2) identical: on input ($x_1\ldots x_n,r_1,\ldots r_n,r'_1,\ldots,r'_2$,R) s.t. $x_1\ldots,x_n$ satisfy R
(1) compute C'_i=Commit($x_i,r_i+r'_i$) and run Prove to get π that values in C'_i satisfy R
(2) compute C_i=Commit(x_i,r_i), run Prove to get π that values in C_i satisfy R, then run Rand

Delegatable Anonymous Credentials



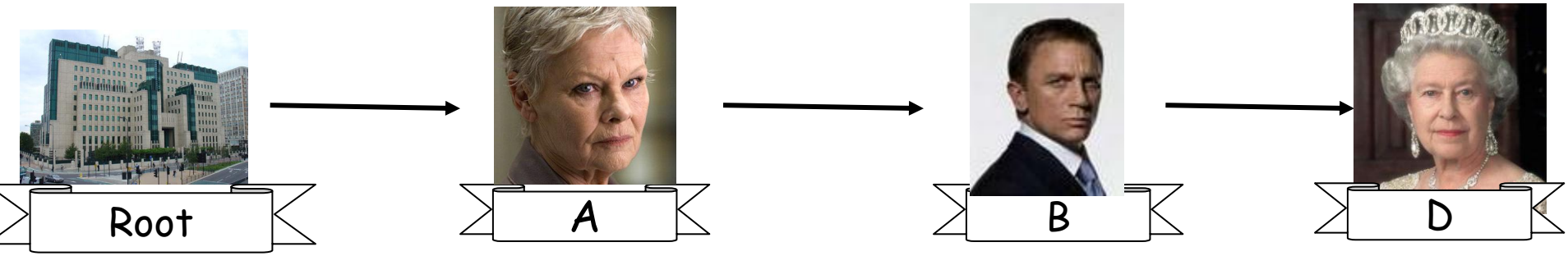
- Each participant has a secret key; $PK_{Root} = Commit(SK_{Root})$
- Root->A credential:
  - A sends to Root: a pseudonym $C_A = Commit(SK_A;r_A)$
  - Root sends to A: proof π_A that C_A was signed under PK_{Root}
  - A's output is $(C_A, \pi_A; r_A)$.

Computed using SPS-Sign and Prove

- A->B credential: (B knows A by $C'_A = Commit(SK_A;r_A+r'_A)$
  - B sends to A: a pseudonym $C_B = Commit(SK_B; r_B)$
  - A sends to B:
    - (1) $\pi'_A = Rand(C_A,\pi,r'_A)$ that C'_A was signed under PK_{Root}
    - (2) π_B that C_B was signed under C'_A
  - B outputs $(C'_A, \pi'_A, C_B, \pi_B; r_B)$

Twist on SPS sigs: Commitments as keys

Delegatable Anonymous Credentials



- How does B delegate to D?
  - D knows B by pseudonym $C'_B = Commit(SK_B; r_B + r'_B)$
  - D sends to B: a pseudonym $C_D = Commit(SK_D; r_D)$
  - B sends to D:
    - (1) $C''_A = C'_A$ e
    - **(2)** $\pi''_A = Ran(\ldots_A, \pi'_A, r''_A)$ that C'' … signed under PK_{Root}
    - **(3)** $\pi'_B = Rand(C'_A, C_B, \pi_B, r''_A, r'_B)$ that C'_B was signed under C''_A
    - (2) π_D that C_D was signed under C'_B
  - D outputs $(C''_A, \pi''_A, C'_B, \pi'_B, C_D, \pi_D ; r_D)$

Computed using Sign a…

Here can also incorporate attributes and e-token info

GS NIWI is Randomizable...

- GS NIZK is randomizable too
- Problem: how to make it “robust,” so that A can’t fake credentials even with access to “simulated” credentials from honest participants?
- Answer: stay tuned for CKLM13!

Conclusion

- NIZK is a practical tool, thanks to GS proofs and bilinear pairings
- Name of the game: express what you want to prove as a PPE

- What we thought was theoretical-only can be practical
- Plus some things (e.g. delegatable credentials) that we didn't even think could be done.