How to build privacy-preserving cryptographic protocols

Jan Camenisch

Principle Researcher; TL Cryptography & Privacy
IBM Research – Zurich

@JanCamenisch
ibm.biz/jancamenisch
33% of cyber crimes, including identity theft, take less time than to make a cup of tea.
Facts

10 Years ago, your identity information on the black market was worth $150. Today….
Facts

$15,000,000,000 cost of identity theft worldwide (2015)
Attackers hide easily in the vast of cyberspace
Houston, we have a problem!
The problem is this…
The Problem is This: Computers Never Forget

- Data is stored by default
- Data mining gets ever better
- Apps built to use & generate (too much) data
- New (ways of) businesses using personal data

- Humans forget most things too quickly
- Paper collects dust in drawers
- But that’s how we design and build applications!
Learnings from Snowden – Take Aways

NSA collects massive amounts of data

*Not* by breaking encryption schemes! But by openness & insecurity of systems, infiltration, ...

- However, Snowden had limited access to docs (no crypt-analysis reports)

Many things doable by ordinary hackers or somewhat sophisticated crooks

- CA compromise
- Stealing data at rest
- Extortion, system manipulations,..

Some things of course require large budget and organization

- FPGA, ASICS
- Deliberate weakening of infrastructure (e.g., PRG standards) – btw, a very bad idea!
Where's all my data?

The ways of data are hard to understand

- Devices, operating systems, & apps are getting more complex and intertwined
  - Mashups, Ad networks
  - Machines virtual and realtime configured
  - Not visible to users, and experts
  - Data processing changes constantly

→ No control over data and far too easy to loose them

IoT makes this even worse!
So it seems beach hurt is rather put on Venus...
Security & Privacy is not a lost cause!

We need paradigm shift:

build stuff for the moon rather than the sandy beach!
That means:

- Use only minimal data necessary
- Encrypt every bit – and keep it like that
- Attach usage policies to each bit

Good news: Cryptography allows for that!
Cryptography to the Aid!

Mix Networks
Oblivious Transfer
Searchable Encryption
Onion Routing
Confermer signatures
Anonymous Credentials
Pseudonym Systems
OT with Access Control
e-voting
Priced OT
Blind signatures
Private information retrieval
Group signatures
Secret Handshakes
Anonymous Credentials
Homomorphic Encryption
Legal side of privacy
Laws and regulations – History

Notion of privacy has changes throughout history (Curtains, shades, etc)

Code of fair information practices (1973)

- Notice/Awareness
- Choice/Consent
- Access/Participation
- Integrity/Security
- Enforcement/Redress
Laws and regulations – History

Many laws follow the same principles

- US Privacy Act 1974
- OECD Guidelines 1980
- EU data protection directive 1995
- General Data Protection Regulation (GDPR) 2018

But: Often only own citizens are protected!

- See, e.g., US laws

Laws are always lagging behind technology, often difficult to capture
OECD Privacy Principles

- **Collection Limitation**
  There should be limits to the collection of personal data and any such data obtained by *lawful and fair* means, and where appropriate, with the *knowledge or consent* of the *data subject*.

- **Data Quality**
  Personal data should be *relevant to the purposes* for which they are to be used, and, to the extend necessary for those purposes, should be *accurate, complete and kept up-to-date*.

- **Purpose Specification**
  The purposes for which personal data are collected should be *specified not later than at the time* of data collection and the subsequent *use limited* to the fulfillment of those purposes.
OECD Privacy Principles

- **Use Limitation**
  Personal data should not be disclosed, made available, or otherwise used for purposes other than those specified under the preceding purpose specification principle
  - except with the consent of the data subject, or
  - by the authority of law.

- **Security Safeguards**
  Personal data should be protected by *reasonable* security safeguards against such risks as *loss or unauthorized access, destruction, use, modification or disclosure*.

- **Accountability**
  A data controller should be accountable for *complying with measures* that give effect to the principles stated above.
OECD Privacy Principles

- **Openness**
  There should be a general policy of openness about developments, practices and policies with respect to personal data.

- **Individual Participation**
  An individual should have the right:
  
  - to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him;
  
  - to have communicated to him, data relating to him
    
    - within a reasonable time,
    
    - at a charge, if any, that is not excessive,
    
    - in a reasonable manner, and Individual Participation in a form that is readily intelligible to him;
  
  - to be given reasons if a request made under subparagraphs (a) and (b) is denied, and to be able to challenge such denial; and

  - to challenge data relating to him and, if the challenge is successful to have the data erased, rectified, completed or amended.
Privacy laws & regulations vary widely throughout the world.

Laws Throughout the World

- European Data Protection Directive
  - privacy = fundamental human right
  - requires all European Union countries to adopt similar comprehensive privacy laws
  - revision in process (higher fines are foreseen)

- US sets on self-regulation, some sector-specific laws, with relatively minimal protections
  - Constitutional law governs the rights of individuals wrt the government
  - Tort (Schadenersatz) law governs disputes between private individuals or other private persons
  - Federal Trade Commission (FTC) deals with consumer protection
    - HIPAA (Health Insurance Portability and Accountability Act, 1996)
    - COPPA (Children’s Online Privacy Protection Act, 1998)
    - GLB (Gramm-Leach-Bliley-Act, 1999)
      → over 7,000 State Legislations & Enforcement activity/Class Actions

- Australia: Privacy Act 1988, Amendment 2012
  - Notify users about collection; Notice if data are sent overseas, stronger access rights

More information http://www.informationshield.com/intprivacylaws.html
The Worst Data Breaches of 2011 -
535 breaches during 2011 that involve 30.4 million sensitive records
(2013: 601 breaches recorded, involving 55 million records)

- **Sony** – after over a dozen data breaches Sony faces class action lawsuits over its failure to protect over 100 million user records.
- **Epsilon** – 60 million customer emails addresses
- **Sutter Physicians Services** – stolen desktop contained 3.3 million patients’ medical details including name, address, phone number, email address & health insurance plan.
- **Tricare and SAIC** – backup tapes were stolen from the car of a Tricare employee; led to a $4.9 billion lawsuit being filed.


And the trend continues :-(
→ Importance of strict privacy & security policies (incl. data retention)
→ Avoid “breaches” simply by properly encrypting sensitive data or, better, using privacy enhancing technologies...
Back to Rocket Science
Privacy & Security by design....
Privacy by design....

@ Communication layer
  - TOR, JAP, etc

@ Authentication layer
  - privacy-preserving attribute-based credentials

@ Application layer
  - eVoting, ePolls, ....
  - all apps should be done as “privacy by design”
Authentication without identification

An example of “rocket science”
Alice wants to watch a movie at Mplex

I wish to see Alice in Wonderland

Movie Streaming Service
Alice wants to watch a movie at Mplex

Alice

Movie Streaming Service

I need proof of:
- subscription
- be older than 12
Watching the movie with the traditional solution

ok, here's
- my eID
- my subscription
Watching the movie with the traditional solution

Aha, you are
- Alice Doe
- born on Dec 12, 1975
- 7 Waterdrive
- CH 8003 Zurich
- Married
- Expires Aug 4, 2018

Mplex Customer
- #1029347
- Premium Subscription
- Expires Jan 13, 2016

Movie Streaming Service
Watching the movie with the traditional solution

This is a privacy and security problem!
- identity theft
- profiling
- discrimination

Aha, you are
- Alice Doe
- born on Dec 12, 1975
- 7 Waterdrive
- CH 8003 Zurich
- Married
- Expires Aug 4, 2018

Mplex Customer
- #1029347
- Premium Subscription
- Expires Jan 13, 2016
Watching the movie with the traditional solution

With OpenID and similar solution, e.g., log-in with Facebook

ok, I'm Alice@facebook.com

Alice

Movie Streaming Service
Watching the movie with the traditional solution

With OpenID and similar solution, e.g., log-in with Facebook

Aha, Alice is watching a 12+ movie
Watching the movie with the traditional solution

With OpenID and similar solution, e.g., log-in with Facebook

Aha, Alice is watching a 12+ movie

Aha, you are
- Alice@facebook.com
- born on Dec 12, 1975
- Alice's friends are ....
- Alice's public profile is ...

Mplex Customer
- #1029347
- Premium Subscription
- Expires Jan 13, 2016

Movie Streaming Service
Authentication – Online Solution

e.g., SAML, WS-Federation, OpenID, Facebook Connect

Privacy: issuer knows who visits which website at which time (often from own records, if not by correlating logs with website)

Security: issuance key online 24/7

1. access request
2. over 18?
3. over 18
4. authenticate
5. over 18
6. over 18
Private Credentials (aka Anonymous Credentials aka Privacy-ABCs)

best of both worlds

- **Privacy:**
  - unlinkable transactions
  - minimal information disclosure

- **Security:**
  - offline issuer

Examples include IBM Identity Mixer and Microsoft U-Prove
With cryptography, we can do much better

When Alice authenticates to the Movie Streaming Service, all the services learns is that Alice

- has a subscription and
- is older than 12

and no more.
Privacy-protecting authentication with IBM Identity Mixer

Like PKI, but better:

- One secret Identity (secret key)
- Many Public Pseudonyms (public keys)
Privacy-protecting authentication with IBM Identity Mixer

Like PKI, but better:

- Issuing a credential

Name = Alice Doe
Birth date = April 3, 1997
Privacy-protecting authentication with Privacy ABCs

Alice

Movie Streaming Service
Privacy-protecting authentication with IBM Identity Mixer

I wish to see Alice in Wonderland

I need proof of:
- subscription
- be older than 12

Movie Streaming Service
Privacy-protecting authentication with IBM Identity Mixer

Alice

Movie Streaming Service
Privacy-protecting authentication with IBM Identity Mixer

Alice

I wish to see Alice in Wonderland

Movie Streaming Service

I need proof of:
- subscription
- be older than 12
Privacy-protecting authentication with IBM Identity Mixer

Like PKI

- but does not send credential
- only minimal disclosure
Privacy-protecting authentication with IBM Identity Mixer

Like PKI
- but does not send credential
- only minimal disclosure

Aha, you are
- older than 12
- have a subscription

Movie Streaming Service

(Public Verification Key of issuer)
Advantages of Identity Mixer

- **For Users: privacy**
  - minimizing disclosure of personal data
  - keeping their identities safe
  - pseudonymous/anonymous access

- **For Service Providers: security, accountability, and compliance**
  - avoiding the risk of loosing personal data if it gets stolen
  - compliance with legislation (access control rules, personal data protection)
  - strong authentication (cryptographic proofs replace usernames/passwords)
  - user identification if required (under certain circumstances)
The Crypto to Realise This and More
Required Technologies

- Signature Schemes
- Encryption Schemes
- Commitment Schemes
- Zero-Knowledge Proofs

..... challenge is to do all this efficiently!
Privacy-protecting authentication with Privacy ABCs

commitment scheme

Alice

signature scheme

zero-knowledge proofs
zero-knowledge proofs
Zero-Knowledge Proofs

- interactive proof between a prover and a verifier about the prover's knowledge

- properties:
  - zero-knowledge
    verifier learns nothing about the prover's secret
  - proof of knowledge (soundness)
    prover can convince verifier only if she knows the secret
  - completeness
    if prover knows the secret she can always convince the verifier
Given group \(<g>\) and element \(y \in <g>\).

Prover wants to convince verifier that she knows \(x\) s.t. \(y = g^x\) such that verifier only learns \(y\) and \(g\).

Prover:
- random \(r\), \(t := g^r\)
- \(s := r - cx\)

Verifier:
- random \(c\)
- \(t = g^s \cdot y^c ?\)

notation: \(PK\{(\alpha): \ y = g^\alpha \} \)
Zero Knowledge Proofs

Proof of knowledge: if a prover successfully convinces a verifier, the secret is extractable.

Prover might do protocol computation in any way it wants & we cannot analyse code.

Thought experiment:

- Assume we have prover as a black box → we can reset and rerun prover
- Need to show how secret can be extracted via protocol interface

\[
t = g^s y^c = g^{s'} y^{c'}
\]

\[
y^{c'-c} = g^{s-s'}
\]

\[
y = g^{(s-s')/(c'-c)}
\]

\[
x = (s-s')/(c'-c) \mod q
\]
Zero Knowledge Proofs: Security

Zero-knowledge property:
If verifier does not learn anything (except the fact that Alice knows $x = \log g y$)

Idea: One can simulate whatever Bob “sees”.

Choose random $c', s'$
compute $t := g^{s'} y^{c'}$
if $c = c'$ send $s' = s$
otherwise restart

Problem: if domain of $c$ too large, success probability becomes too small
Zero Knowledge Proofs of Knowledge of Discrete Logarithms

One way to modify protocol to get large domain $c$:

Prover: $r \rightarrow t := g^r \rightarrow h := H(c,v) \rightarrow s := r - cx$

Verifier: $y, g \rightarrow h := H(c,v) \rightarrow c,v \rightarrow t = g^s y^c$?

notation: $PK\{(a): y = g^a \}$
Zero Knowledge Proofs: Security

One way to modify protocol to get large domain $c$:

Choose random $c',s'$
compute $t' := g^{s'} y^{c'}$

after having received $c$ “reboot” verifier

Choose random $s$
compute $t := g^s y^c$

send $s$
Zero Knowledge Proofs of Knowledge of Discrete Logarithms

Given group \(<g>\) and element \(y \in <g>\).

Prover wants to convince verifier that she knows \(x\) s.t. \(y = g^x\) such that verifier only learns \(y\) and \(g\).

\[
\begin{align*}
\text{Prover:} & \quad \text{Verifier:} \\
\text{random } r & \quad y, g \\
t & := g^r & t = g^s y^c \quad \text{?} \\
s & := r - cx & \\
\text{notation: } PK\{(a): \ y = g^a\}
\end{align*}
\]
From Protocols To Signatures

Signing a message $m$:
- choose random $r \in \mathbb{Z}_q$ and
- compute $c := H(g^r || m) = H(t || m)$
  
  $s := r - cx \mod (q)$

- output $(c, s)$

Verifying a signature $(c, s)$ on a message $m$:
- check $c = H(g^s y^c || m) \ ? \iff t = g^s y^c \ ?$

Security:
- underlying protocol is zero-knowledge proof of knowledge
- hash function $H(.)$ behaves as a “random oracle.”
Zero Knowledge Proofs of Knowledge of Discrete Logarithms

Many Exponents:

$$\text{PK}\{ (\alpha, \beta, \gamma, \delta) : \ y = g^\alpha h^\beta z^\gamma k^\delta u^\beta \}$$

Logical combinations:

$$\text{PK}\{ (\alpha, \beta) : \ y = g^\alpha \land z = g^\beta \land u = g^\beta h^\alpha \}$$
$$\text{PK}\{ (\alpha, \beta) : \ y = g^\alpha \lor z = g^\beta \}$$

Intervals and groups of different order (under SRSA):

$$\text{PK}\{ (\alpha) : \ y = g^\alpha \land \alpha \in [A, B] \}$$

Non-interactive (Fiat-Shamir heuristic $\rightarrow$ Schnorr Signatures):

$$\text{SPK}\{ (\alpha) : \ y = g^\alpha \}(m)$$
Some Example Proofs and Their Analysis

Let \( g, h, C_1, C_2, C_3 \) be group elements.

Now, what does the following mean?

\[
\text{PK}\{(a_1, b_1, a_2, b_2, a_3, b_3) : \quad C_1 = g^{a_1} h^{b_1} \land \quad C_2 = g^{a_2} h^{b_2} \land \quad C_3 = g^{a_3} h^{b_3} \land \quad C_3 = g^{a_1 + a_2} h^{b_3}\}
\]

\[\rightarrow\text{Prover knows values } a_1, b_1, a_2, b_2, b_3 \text{ such that}\]

\[C_1 = g^{a_1} h^{b_1}, \quad C_2 = g^{a_2} h^{b_2} \quad \text{and} \quad C_3 = g^{a_1 + a_2} h^{b_3} = g^{a_3} h^{b_3}\]

\[\rightarrow a_3 = a_1 + a_2 \pmod{q}\]

And what about:

\[
\text{PK}\{(a_1, \ldots, b_3) : \quad C_1 = g^{a_1} h^{b_1} \land \quad C_2 = g^{a_2} h^{b_2} \land \quad C_3 = g^{a_3} h^{b_3} \land \quad C_3 = g^{a_1 + 5a_2} h^{b_3}\}
\]

\[\rightarrow C_3 = g^{a_1 + 5a_2} h^{b_3} = g^{a_1 + 5a_2} h^{b_3}\]

\[\rightarrow a_3 = a_1 + 5a_2 \pmod{q}\]
Some Example Proofs and Their Analysis

And \( PK\{ (\alpha_1, \ldots, \beta_4) : \ C_1 = g^{\alpha_1} h^{\beta_1} \land C_2 = g^{\alpha_2} h^{\beta_2} \land C_3 = g^{\alpha_3} h^{\beta_3} \land C_3 = C_2^{\alpha_1} h^{\beta_4} \} \)?

\[ \rightarrow \text{Prover knows values } \alpha_1, \beta_1, \alpha_2, \beta_2, \beta_3 \text{ such that} \]

\[ C_3 = C_2^{\alpha_1} h^{\beta_4} = (g^{\alpha_2} h^{\beta_2})^{\alpha_1} h^{\beta_4} = g^{\alpha_2 \cdot \alpha_1} h^{\beta_4 + \beta_2 \cdot \alpha_1} \]

\[ C_3 = g^{\alpha_2 \cdot \alpha_1} h^{\beta_3 + \beta_2 \cdot \alpha_1} = g^{\alpha_3} h^{\beta_3} \]

\[ \rightarrow a_3 = a_1 \cdot a_2 \pmod{q} \]
Let $g, h, C1, C2, C3$ be group elements.

Some Example Proofs and Their Analysis

Now, what does

$$PK\{(a_1,..,\beta_2): \quad C1= g^{a_1}h^{\beta_1} \land C2= g^{a_2}h^{\beta_2} \land g = (C2/C1)^{a_1}h^{\beta_2} \}$$

mean?

→ Prover knows values $\alpha, \beta_1, \beta_2$ such that

$$C1= g^{a_1}h^{\beta_1}$$

$$g = (C2/C1)^{a_1}h^{\beta_2} = (C2 g^{-a_1}h^{-\beta_1})^{a_1}h^{\beta_2}$$

→

$$g^{1/a_1} = C2 g^{-a_1}h^{-\beta_1}h^{\beta_2/a_1}$$

$$C2 = g^{a_1}h^{\beta_1}h^{-\beta_2/a_1}g^{1/a_1} = g^{a_1 + 1/a_1}h^{\beta_1-\beta_2/a_1}$$

$$C2 = g^{a_2}h^{\beta_2}$$

$$a_2 = a_1 + a_1^{-1} \pmod{q}$$
signature schemes
Signature Scheme: Functionality

Key Generation
Signature Scheme: Functionality

\[ \sigma = \text{sig}((m_1, \ldots, m_k)) \]
Signature Scheme: Functionality

\[ \sigma = \text{sig}(m_1, \ldots, m_k) \]

Verification

\[ \text{ver}(\sigma, (m_1, \ldots, m_k), \varphi) = \text{true} \]
Signature Scheme: Security

Unforgeability under Adaptive Chosen Message Attack

\[ m_1 \rightarrow \sigma_1 \rightarrow \text{Adversary} \]
Signature Scheme: Security

Unforgeability under Adaptive Chosen Message Attack
Signature Scheme: Security

Unforgeability under Adaptive Chosen Message Attack

σ₁ and m₁ ≠ mᵢ s.t.

\text{ver}(\sigma', m', \varnothing) = \text{true}
Signature Scheme: Security

Unforgeability under Adaptive Chosen Message Attack

\[ \sigma_1 \]

\[ m_1 \]

\[ \sigma \]

\[ m_i \]

\[ \text{ver}(\sigma', m', \text{true}) = \text{true} \]

\[ \sigma' \text{ and } m' \neq m_i \]
RSA Signature Scheme – for reference

Rivest, Shamir, and Adlemann 1978

Secret Key: two random primes $p$ and $q$

Public Key: $n := pq$, prime $e$, and collision-free hash function $H: \{0,1\}^* \rightarrow \{0,1\}^\ell$

Computing signature on a message $m \in \{0,1\}^*$

\[ d := \frac{1}{e} \mod (p-1)(q-1) \]

\[ s := H(m)^d \mod n \]

Verification of signature $s$ on a message $m \in \{0,1\}^*$

\[ s^e = H(m) \mod n \]

Correctness:

\[ s^e = (H(m)^d)^e = H(m)^{d\cdot e} = H(m) \mod n \]
RSA Signature Scheme – for reference

Verification signature on a message \( m \in \{0,1\}^* \)

\[ s^e := H(m) \quad (\text{mod } n) \]

Wanna do proof of knowledge of signature on a message, e.g.,

\[ \text{PK}\{ (m,s): s^e = H(m) \quad (\text{mod } n) \} \]

But this is not a valid proof expression!!!! :-(
CL-Signature Scheme

Public key of signer: RSA modulus $n$ and $a_i, b, d \in \mathbb{QR}_n$

Secret key: factors of $n$

To sign $k$ a $m_1, \ldots, m_k \in \{0,1\}^\ell$:

- choose random prime $2^{\ell+2} > e > 2^{\ell+1}$ and integer $s \approx n$
- compute $c$:

$$c = \left( d / \left( a_1^{m_1} \cdots a_k^{m_k} b^s \right) \right)^{1/e} \mod n$$

- signature is $(c,e,s)$
To verify a signature \((c,e,s)\) on messages \(m_1, \ldots, m_k\):

- \(m_1, \ldots, m_k \in \{0,1\}^\ell\):
- \(e > 2^{\ell+1}\)
- \(d = c^e a_1^{m_1} \cdots a_k^{m_k} b^s \mod n\)

Theorem: 
Signature scheme is secure against adaptively chosen message attacks under Strong RSA assumption.
Proving Knowledge of a CL-signature

Recall: \[ d = c^e a_1^{m_1} a_2^{m_2} b^s \mod n \]

Observe:
- Let \( c' = c b^t \mod n \) with randomly chosen \( t \)
- Then \( d = c'^e a_1^{m_1} a_2^{m_2} b^{s-et} \mod n \), i.e., \((c',e,s^* = s-et)\) is also signature on \( m_1 \) and \( m_2 \)

To prove knowledge of signature \((c',e,s^*)\) on \( m_2 \) and some \( m_1 \)
- provide \( c' \)
- \( \operatorname{PK}\{(\varepsilon, \mu_1, \sigma) : d/a_2^{m_2} := c'^\varepsilon a_1^{\mu_1} b^\sigma \land \mu \in \{0,1\}^\ell \land \varepsilon > 2^{\ell+1} \} \)

\[ \rightarrow \text{proves } d = c'^\varepsilon a_1^{\mu_1} a_2^{m_2} b^\sigma \]
Privacy-protecting authentication with Privacy ABCs

commitment scheme

Alice

signature scheme

zero-knowledge proofs
commitment scheme
Commitment Scheme: Functionality

- A person sends a message \( m \) to a lock.
- The lock contains \( m \).
- The other person asks if \( m \) is the correct message, \( m \in \mathbb{m} \).

Note: The image includes a message \( m, 2-36-17 \) which is not directly related to the commitment scheme functionality described.
Commitment Scheme: Security

Binding

m, 2-36-17

m', 3-21-11

m, m' ∈ m
Commitment Scheme: Security

Binding

m, 2-36-17

m, 3-21-11

m', ?

m, ?
Commitment Scheme: Security

Hiding: for all message $m, m'$
Hiding: for all message $m, m'$

Commitment Scheme: Security
Commitment Schemes

Group $G = \langle g \rangle = \langle h \rangle$ of order $q$

To commit to element $x \in \mathbb{Z}_q$:

- **Pedersen**: perfectly hiding, computationally binding
  
  choose $r \in \mathbb{Z}_q$ and compute $c = g^x h^r$

- **ElGamal**: computationally hiding, perfectly binding:
  
  choose $r \in \mathbb{Z}_q$ and compute $c = (g^x h^r, g^r)$

To open commitment:

- reveal $x$ and $r$ to verifier
- verifier checks if $c = g^x h^r$
Pedersen's Commitment Scheme

Pedersen's Scheme: Choose $r \in \mathbb{Z}_q$ and compute $c = g^x h^r$

Perfectly hiding:

Let $c$ be a commitment and $u = \log_g h$

Thus $c = g^x h^r = g^{x + ur} = g^{(x + ur') + u(r - r')}$

$$= g^{x + ur'} h^{r - r'}$$ for any $r'$!

I.e., given $c$ and $x'$ there always exist $r'$ such that $c = g^{x'} h^{r'}$

Computationally binding:

Let $c$, $(x', r')$ and $(x, r)$ s.t. $c = g^{x'} h^{r'} = g^x h^r$

Then $g^{x' - x} = h^{r - r'}$ and $u = \log_g h = (x' - x)/(r - r') \mod q$
Commitment Scheme: Extended Features

Proof of Knowledge of Contents

Proof of Relations among Contents
Proof of Knowledge of Contents

Let $C_1 = g^m h^r$ and $C' = g^{m'} h^r$ then:

\[ \text{Proof of Knowledge of Contents} \]

\[ \text{Proof} \]

\[ \text{true} \]

\[ \text{PK}\{(\alpha, \beta): \ C = g^\beta h^\alpha \} \]

Proof of Relations among Contents

\[ \text{Proof} \]

\[ \text{true} \]

\[ \text{PK}\{(\alpha, \beta, \gamma): \ C' = g^\beta h^\alpha \land C = (g^2)^\beta h^\gamma \} \]
Protocol building frameworks

- Discrete log-based
  - CL-signatures (RSA,ECC), Schnorr proofs, Pedersen Commitment
  - CS-Verifiable Encryption
- Structure preserving primitives tailored to Groth-Sahai proofs
- Quantum Safe / Latticed-based crypto: not quite there yet

... Haven’t talked about modeling and proving security at all
- UC-framework
- Constructive Crypto
- IITM
putting things together
Realizing Pseudonyms and Key Binding

- Let $G = \langle g \rangle = \langle h \rangle$ of order $q$
- User's secret key: random $sk \in \mathbb{Z}_q$
- To compute a pseudonym $Nym$
  - Choose random $r \in \mathbb{Z}_q$
  - Compute $Nym = g^{sk} h^r$
Privacy-protecting authentication with Privacy ABCs

Like PKI, but better:

- Issuing a credential

Name = Alice Doe
Birth date = April 3, 1997

Concept: credentials
Realizing Issuance of Credential

Recall: a signature $(c,e,s)$ on messages $m_1, \ldots, m_k$:

- $m_1, \ldots, m_k \in \{0,1\}^\ell$:
- $e > 2^{\ell+1}$
- $d = c^e a_1^{m_1} \cdots a_k^{m_k} b^s \mod n$

Problem: Pseudonym not in message space!

Solution: Sign secret key instead

\[ d = c^e a_1^{sk} a_2^{m_2} \cdots a_k^{m_k} b^s \mod n \]

New Problem: how can we sign a secret message?
Verification remains unchanged!
Security requirements basically the same as for signatures, but

- signer should not learn any information about $m_1, ..., m_j$
- Forgery w.r.t. message clear parts and opening of commitments
Realizing Issuance of Credential

\[ C = a_1^{sk} b^{s'} \]
Realizing Issuance of Credential

\[ PK((\mu_1, \sigma')): C = a_1^{\mu_1} b^{\sigma'} \]

\[ C = a_1^{sk} b^{s'} \]
Realizing Issuance of Credential

$C = a_1^{sk} b^{s'}$

$PK((\mu_1, \sigma')) : C = a_1^{\mu_1} b^{\sigma'}$

$c = (d/C a_2^{name} b^{s''})^{1/e} \mod n$

$C, name$

$(c, e, s'')$
Realizing Issuance of Credential

\[ C = a_1^{sk} b^{s'} \]
\[ C, \text{name} \]
\[ c = (d/C a_2^{\text{name}} b^{s''})^{1/e} \mod n \]
\[ d = c^e a_1^{sk} a_2^{\text{name}} b^{s''} + s' \mod n \]
Realizing Issuance of Credential

Want to sign w.r.t. $Nym = g^{sk}h^r$
Realizing Issuance of Credential

Want to sign w.r.t. $Nym = g^{sk} h^r$

$Nym = g^{sk} h^r$

$C = a_1^{sk} a_2^r b^{s'}$

$C, Nym, name$

$n, a_i, b, d$

stores $Nym, name$

$c = (d/C a_3^{name} b^{s''})^{1/e} \mod n$

$d = c e^{a_1^{sk} r}^{a_2 a_3^{name} b^{s''} + s'} \mod n$
An Example Scenario
Polling: Scenario and Requirements

Scenario:

- Pollster(s) and a number of users
- Only registered user (e.g., students who took a course) can voice opinion (e.g., course evaluation)
- User can voice opinion only once (subsequent attempts are dropped)
- Users want to be anonymous
- A user's opinion in different polls must not be linkable
Polling – Solution: Registration

- User generates pseudonym (ID for registration)
- User obtains credential on pseudonym stating that she is eligible for polls, i.e., \((c,e,s)\)
  \[
d = c^e a_1^{sk} a_2^r a_3^{attr} b^s \pmod{n}
\]
- Credential can contain attributes (e.g., course ID) about her
Polling – Solution: Submit Poll

1. User generates domain pseudonym, domain = pollID

2. User transforms credential

3. Transformed credential with a subset of the attributes
   - User is anonymous and unlinkable
   - Multiple opinions are detected because uniqueness of domain pseudonym
Polling – Solution: Polling

1. Domain pseudonym: \( P = g_d^{sk} = H(\text{pollID})^{sk} \)

\[ P_1 = H(\text{pollID1})^{sk} \quad \text{and} \quad P_2 = H(\text{pollID2})^{sk} \]
are unlinkable

(under the Decisional Diffie-Hellman assumption & random oracles)

2. User transforms credential:

- \( c' = c \cdot b^{s'} \mod n \) with randomly chosen \( s' \)

- \( \text{SPK}\{ (\varepsilon, \mu_1, \mu_2, \mu_3, \sigma) : \ P = g_d^{\mu_1} \land d := c' \cdot \varepsilon \cdot a_1^{\mu_1} \cdot a_2^{\mu_2} \cdot a_3^{\mu_3} \cdot b^{\sigma} \mod n \land \mu_1, \mu_2, \mu_3 \in \{0,1\}^\ell \land \varepsilon > 2^{\ell+1} \} \)(opinion)
There is more
Here's just a few examples
• If car is broken: ID with insurance needs be retrieved
• Can verifiably encrypt any certified attribute (*optional*)
• TTP is off-line & can be distributed to lessen trust
If Alice was speeding, license needs to be revoked!

There are many different use cases and many solutions
  * Variants of CRL work (using crypto to maintain anonymity)
    * Accumulators
    * Signing entries & Proof, ....
  * Limited validity – certs need to be updated
  * ... For proving age, a revoked driver's license still works
Limits of anonymity possible (*optional*):
- If Alice and Eve are on-line together they are caught!
- Use Limitation – anonymous until:
  - If Alice used certs > 100 times total...
  - ... or > 10'000 times with Bob
- Alice's cert can be bound to hardware token (e.g., TPM)
Some Use Cases
Age verification

Proving 12+, 18+, 21+ without disclosing the exact date of birth – privacy and compliance with age-related legislation

- Movie streaming services
- Gaming industry
- Online gambling platforms
- Dating websites
- Social benefits for young/old people
Use Case: Anonymous Authentication

Use certificate anonymously
Use Case: Anonymous Authentication

Use certificate anonymously
Server shall be unable to tell whether or not it is the same user
Use Case: Anonymous Access to a Database

Simple case: DB learns not who accesses DB
Better: Oblivious Access to Database (OT with AC)

- Server must not learn *who* accesses
- *which* record
- Still, Alice can access only records she is *authorized* for
Subscriptions, membership

Who accesses *which data* at which time can reveal sensitive information about the users (their research strategy, location, habits, etc.)

- Patent databases
- DNA databases
- News/Journals/Magazines
- Transportation: tickets, toll roads
- Loyalty programs
Use case: Direct Anonymous Attestation

- Attestation of computer state by TPM (root of trust)
- TPM measures boot sequence
- TPM attest boot sequence to third party
- Attestation based on cryptographic keys
- Privacy needs to be protected → Different transaction must not be linkable

→ Requires strong authentication of TPM but such that privacy of users and data be protected
Direct Anonymous Attestation (Brickell, Camenisch, Chen - 2003)

Two crucial differences:

1. One secret key - several public keys

2. Randomizable credentials: original credential into new credentials that “looks like” a fresh credential
   → different randomize credentials cannot be linked (anonymity)
   → still credentials are unforgeable
Some context on DAA

- RSA-based scheme standardized by TCG in 2004, later also in ISO
- Replaced by ECC-based scheme in 2015 (both TCG and ISO)
- DAA algorithm is split in TPM and host part, ECC-based scheme only defined for TPM

- New scheme supports multiple ECC-based DAA protocols (q-SDH, LRSW based etc)
- Scheme is really efficient: TPM computes single exponentiation only
- FIDO authenticator based on TPM DAA specified
How the TPM and the Host Sign Jointly (simplyfied)

$\text{PK}\{ (x, r) : y = g^x h^z \}$

$\text{PK}\{ (x) : y' = g^x \}$
How the TPM and the Host Sign Jointly (simplified)

\[ PK\{ (x, m) : y = g^{xh^z} \} \]

\[ PK\{ (x) : y' = g^x \} \]

random \( r_1 \)

\( t' = g^{r_1} \)

\( t' = g^{r_1} \)

random \( r_2 \)

\( t = t'h^{r_2} \)

\( t = t'h^{r_2} \)
How the TPM and the Host Sign Jointly (simplified)

\[

PK\{(x, m) : y = g^x h^z\}
\]

\[

PK\{(x) : y' = g^x\}
\]

random \( r_1 \)

\( t' = g^{r_1} \)

c

t' \rightarrow \text{random} \ r_2

t = t' h^{r_2}

c

random \ c
How the TPM and the Host Sign Jointly (simplyfied)

\[
PK\{ (x, m) : y = g^x h^z \}
\]

random \( r_1 \)

\( t' = g^{r_1} \)

\[
s_1 = r_1 - c x
\]

t = \( t'h^{r_2} \)

\[
s_2 = r_2 - c z
\]

\[
t = y^c g^{s_1} h^{s_2}
\]
How the TPM and the Host Sign Jointly (simplified)

\[ t' = g^{r_1} \]
\[ t' = g^{r_1} \]
\[ c = r_1 \cdot x \]
\[ s_1 = r_1 - c \cdot x \]

\[ t = t'^{h^{r_2}} \]
\[ t = t'^{h^{r_2}} \]
\[ t = t'^{h^{r_2}} \]
\[ c = r_2 - c \cdot z \]
\[ s_2 = r_2 - c \cdot z \]

\[ PK\{ (x, m) : y = g^x \} \]
\[ PK\{ (x, m) : y = g^x \} \]
\[ PK\{ (x, m) : y = g^x \} \]
\[ PK\{ (x) : y' = g^x \} \]

\[ PK\{ (x) : y' = g^x \} \]
\[ PK\{ (x) : y' = g^x \} \]
\[ PK\{ (x) : y' = g^x \} \]

TPM spec

not spec'ed (was TSS spec)
Healthcare Use Case

Anonymous consultations with specialists

- online chat with at physician / online consultation with IBM Watson to check eligibility

1. Alice proves she has insurance
2. Alice describes symptoms
3. Alice gets credential that she is allowed to get treatment
4. Alice gets treatment from physician, hospital, etc
5. Alice sends bill to insurance and proves that she had gotten the necessary permission for the treatment.
Payment Use Case

- Credential = Bank note
- Double spending need to be prevented/detected
  - On-line or Off-line modi possible
- Money laundering can also be taken care of
Realizing On-Line eCash
eCash scenario & requirements

Requirements
- **Anonymity**: Withdrawal and Deposit must be unlinkable
- **No Double Spending**: Coin is bit-strings, can be spend twice
Towards a Solution: do it like paper money

Sign notes with digital signature scheme

- Note = (serial number #, value)
- Secure because
  - signature scheme can not be forged
  - bank will accepts some serial number only once → on-line e-cash
- Not anonymous because (cf. paper solution)
  - bit-string of signature is unique
  - serial number is unique
Towards a Solution → on-line ecash

- Use (more) cryptography
  - Hide serial number from bank when issuing
    - e.g., sign commitment of serial number
  - Reveal serial number and proof
    - knowledge of signature on
    - commitment to serial number
  - Anonymous because of commitments scheme and zero-knowledge proof
Security

Anonymity

- Bank does not learn # during withdrawal
- Bank & Shop do not learn c, e when spending

Unforgeability

- User can use serial number only once
Realizing Off-Line eCash
Making e-cash offline – basic issue

On-Line Solution:
1. Coin = random serial # (chosen by user) signed by Bank
2. Banks signs blindly
3. Spending by sending # and prove knowledge of signature to Merchant
4. Merchant checks validity w/ Bank
5. Bank accepts each serial # only once.

Off-Line:
- Can check serial # only after the fact
- … but at that point user will have been disappeared...
Towards off-line signatures

Goal:
– spending coin once: OK
– spending coin twice: anonymity is revoked

Seems like a paradox, but crypto is all about solving paradoxical problems :-)

Main Idea:

- Include $\#$, id, $r$ in credential ($\#$, $r$ hidden from bank/issuer)
- Upon spending:
  
  reveal $\#$ and $t = \text{id} + r \cdot u$

  with $u$ randomly chosen by merchant and prove relation

  $t$ won't reveal anything about id!

- However, given two (or more) equations (for the same $\#$, id, $r$)
  
  $t1 = \text{id} + r \cdot u1$
  
  $t2 = \text{id} + r \cdot u2$

  one can solve for id.
Off-line E-cash: Withdrawal

choose random \(\#, r, s'\) and compute

\[ C = a_1^\# a_2^r b^{s'} \]

\[ d = c^e C a_3^{nym} b^{s''} \pmod{n} \]

\((c,e,s''+s')\) s.t. \(d = c^e a_1^\# a_2^r a_3^{nym} b^{s''+s'} \pmod{n}\)
Off-line E-cash: Payment

compute

\[ t = r + u \text{nym} \mod q \]

\[ c' = c b^{s'} \mod n \]

proof = PK\{(ε, μ, ρ, σ) : \]

\[ d / a_1^# = c'ε a_2^ρ a_3^μ b^σ \pmod n \land g^t = g^ρ (g^u^μ) \} \]
Off-line E-cash: Payment

PK\{(\varepsilon, \mu, \rho, \sigma) : \quad d / a_1^\# = c' \varepsilon a_2^\rho a_3^\mu b^\sigma \pmod{n} \land g^\dagger = g^\rho (g^\mu)^\mu \}\n
1. \quad d = c' \varepsilon a_1^\# a_2^\rho a_3^\mu b^\sigma \pmod{n} \\
   \Rightarrow (c', \varepsilon, \sigma) \text{ is a signature on } (\#, \mu, \rho)

2. \quad g^\dagger = g^{\rho + u \mu} \Rightarrow \dagger = \rho + u \mu \pmod{q}, \text{ i.e., } \dagger \text{ was computed correctly!}
If \( \# \in L \)?

1. \( t = \rho + u \mu \pmod{q} \)
2. \( t' = \rho + u' \mu \pmod{q} \)

solve for \( \rho \) and \( \mu \).

=> \( \mu = nym \) because of proof
Off-line E-cash: Security

- **Unforgeable:**
  - no more coins than \#'s,
    - otherwise one can forge signatures
    - or proofs are not sound
  - if coins with same \# appears with different u's => reveals \textit{nym}

- **Anonymity:**
  - \# and \( r \) are hidden from signer upon withdrawal
  - \( t \) does not reveal anything about \textit{nym} (is blinded by \( r \))
  - proof \textit{proof} does not reveal anything
Extensions

- K-spendable cash
  - Multiple serial numbers and randomizers per coin
  - Use PRF to generate serial number and randomizers from seed in coin
- Money laundering preventions
  - Must not spend more that $10000 dollars with same party
  - Essentially use additional coin defined per merchant that controls this
Conclusion
Conclusions

Roadmap
- Explain possibilities to engineers, policy makers etc
- Usable prototypes
- Public infrastructure for privacy protection

Challenges
- Internet services get paid with personal data (inverse incentive)
- End users are not able to handle their data (user interfaces..)
- Security technology typically invisible and hard to sell
- Solutions seem easier without privacy and security

Towards a secure information society
- Society changes quickly and gets shaped by technology
- Consequences are hard to grasp (time will show...)
- We must inform and engage in a dialog
Let's do some rocket science together!

For information:
- www.zurich.ibm.com/idemix
- idemixdemo.mybluemix.net

jca@zurich.ibm.com    @JanCamenisch    ibm.biz/JanCamenisch
Thank you!

▪ eMail: jca@zurich.ibm.com

▪ Links:
  – www.abc4trust.eu
  – www.PrimeLife.eu
  – www.zurich.ibm.com/idemix
  – github.com/p2abcengine & abc4trust.eu/idemix
Securing Credit Card Payments

The credit card data is never revealed to the merchant, only to the credit card provider

- Bank issues a classic credit card
- User registers at a special portal to obtain the Identity Mixer credential
- User derives a token allowing that store to withdraw the money
- Users cannot be linked across purchases/shops
- Stored credit card info useless to hackers!
Use case: Direct Anonymous Attestation

- Attestation of computer state by TPM (root of trust)
- TPM measures boot sequence
- TPM attest boot sequence to third party
- Attestation based on cryptographic keys
- Privacy needs to be protected → Different transaction must not be linkable

→ Requires strong authentication of TPM but such that privacy of users and data be protected
Attestation extended

Many other different use case: any IoT, Industry 4.0, Home Appliances, Metering, ...
Attempt at Solving Attestation

Problem: using traditional certificates, all transactions of the same platform become linkable :-(

Not Rocket Science!
Direct Anonymous Attestation (Brickell, Camenisch, Chen - 2003)

Two crucial differences:
1. One secret key - several public keys
Two crucial differences:

1. One secret key - several public keys
Direct Anonymous Attestation (Brickell, Camenisch, Chen - 2003)

Two crucial differences:

1. One secret key - several public keys

2. Randomizable credentials: original credential into new credentials that “looks like” a fresh credential
   → different randomize credentials cannot be linked (anonymity)
   → still credentials are unforgeable
Status DAA 2017

- RSA-based scheme standardized by TCG in 2004, later also in ISO
- Replaced by ECC-based scheme in 2015 (both TCG and ISO)
- DAA algorithm is split in TPM and host part, ECC-based scheme only defined for TPM

- New scheme supports multiple ECC-based DAA protocols (q-SDH, LRSW based etc)
- Scheme is really efficient: TPM computes single exponentiation only
- We’ve identified some security issues, these got fixed in latest TPM spec
- See our paper at IEEE S&P 2017 with full specifications and security proof
- FIDO authenticator based on TPM DAA specified
Conclusions

- Device authentication more relevant than ever
- Data parsimony is key to obtain security
- Fancy crypto can realize this, today!
- More public awareness and discussion needed
Let’s do some rocket science together!

For information:

- www.zurich.ibm.com/idemix
- idemixdemo.mybluemix.net

jca@zurich.ibm.com   @JanCamenisch   ibm.biz/JanCamenisch