We all increasingly conduct our daily tasks electronically

....are becoming increasingly vulnerable to cybercrimes
33% of cyber crimes, including identity theft, take less time than to make a cup of tea.
10 Years ago, your identity information on the black market was worth $150. Today....
$4'500'000'000 cost of identity theft worldwide
Houston, we have a problem!
Houston, we have a problem!

“Buzz Aldrin's footprints are still up there”
(Robin Wilton)
Computers don't forget

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

- Humans forget most things too quickly
- Paper collects dust in drawers
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
Applications are designed with the sandy beach in mind but are then built on the moon.

- Feature creep, security comes last, if at all
- Everyone can do apps and sell them
- Networks and systems hard not (well) protected
We need paradigm shift & build stuff for the moon rather than the sandy beach!
That means:

- Reveal only minimal data necessary
- Encrypt every bit
- Attach usage policies to each bit

Cryptography can do that!
Cryptography to the Aid
Today: two solutions

Identity mixer: privacy protecting authentication

Password-based security: from humans to cryptographic keys
Identity Mixer
Alice wants to watch a movie at Movie Streaming Service

I wish to see Alice in Wonderland
Alice wants to watch a movie at Movie Streaming Service

You need:
- subscription
- be older than 12

Alice

Movie Streaming Service
Watching the movie with the traditional solution

Using digital equivalent of paper world, e.g., with X.509 Certificates

ok, here's
- my eID
- my subscription

Alice

Movie Streaming Service
Watching the movie with the traditional solution

...with X.509 Certificates

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
- discrimination
- profiling, possibly in connection with other services

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

With OpenID (similar protocols), 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, you are
- Alice@facebook.com
- 12+
Mplex Customer
- #1029347
- Premium Subscription
- Expires Jan 13, 2016

Aha, Alice is watching a 12+ movie

Alice

Movie Streaming Service
Proper cryptography solves this: Identity Mixer

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

- has a subscription
- is older than 12

and no more!
Privacy-protecting authentication with Privacy ABCs

Users' Keys:

- One secret Identity (secret key)
- Many Public Pseudonyms (public keys)

→ use a different identity for each communication partner or even transaction
Certified attributes from Identity provider

- Issuing a credential

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

Certified attributes from purchasing department

- Issuing a credential
Privacy-protecting authentication with Privacy ABCs

I wish to see Alice in Wonderland

You need:
- subscription
- be older than 12
Proving identity claims

- but does *not* send credentials
- only minimal disclosure

- valid subscription
- eID with age ≥ 12
Proving Identity Claims: Minimal Disclosure

Alice Doe
Dec 12, 1998
Hauptstr. 7, Zurich
CH
single
Exp. Aug 4, 2018

Age: 12+
Exp. Valid
Privacy-protecting authentication with Privacy ABCs

Proving identity claims
- but does not send credential
- only minimal disclosure

Transaction is not linkable to any other of Alice's transactions!

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

(Public Verification Key of issuer)
Try Identity Mixer for yourself

Try yourself:
Build your app:
Source code:
Info:

→ idemixdemo.mybluemix.net
→ github.com/IBM-Bluemix/idemix-issuer-verifier
→ github.com/github.com/p2abcengine/p2abcengine
→ ibm.biz/identity_mixer
Identity Mixer (and related protocols) in standards

- FIDO Alliance authentication is standardizing this as well (w/ and w/out chip)

TPMs allow one to store secret key in a secure place!
Other examples: secure and privacy access to databases

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

- DNA databases
- News/Journals/Magazines
- Patent database

Cryptography access protocol s.t. database provider has *no* information about

- which user accesses
- which data
A glimpse at the underlying cryptography
A Glimpse at the technical realization

Signature scheme compatible with ZKP

Commitment scheme compatible with ZKP & sig. scheme

Zero knowledge proof of knowledge
Given group \( <g> \) and element \( y \in <g> \).
Prover wants to convince verifier that she knows \( x = \log g y \) such that verifier only learns \( y \) and \( g \).

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

Verifier:
- Random \( c \)
- \( t = g^s y^c \)

Proof:
- PK\((\alpha): \ y = g^\alpha \)
Many Exponents:

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

Logical combinations:

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

Intervals and groups of different order (under SRSA):

\[ \text{PK}\{ (\alpha) : \quad y = g^\alpha \land \alpha \in [A,B] \} \]
\[ \text{PK}\{ (\alpha) : \quad y = g^\alpha \land z = g^\alpha \land \alpha \in [0, \min\{\text{ord}(g),\text{ord}(g)\}] \} \]

Non-interactive (Fiat-Shamir heuristic, Schnorr Signatures):

\[ \text{PK}\{ (\alpha) : \quad y = g^\alpha \} (m) \]
RSA Signature Scheme

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 \( m \in \{0,1\}^* \)
\[
s^e = H(m) \pmod{n}
\]

Correctness: \( s^e = (H(m)^d)^e = H(m)^{d^e} = H(m) \pmod{n} \)
RSA Signature Scheme

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

$$s^e := H(m) \pmod{n}$$

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

$$PK\{(m,s): s^e = H(m) \pmod{n}\}$$

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

Public key of signer: RSA modulus $n$ and $a_i, b, d \in QR_n$.

Secret key: factors of $n$

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

• choose random prime $2^{\ell+2} > e > 2^{\ell+1}$ and integer $s \approx n$

• compute $c$:

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

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

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

**Theorem:** Signature scheme is secure against adaptively chosen message attacks under Strong RSA assumption.
Observe:

- \( d = c^e a^m b^s \mod n \)
- Let \( c' = c b^\dagger \mod n \) with randomly chosen \( \dagger \)
  then \( d = c'^e a_1^{m_1} a_2^{m_2} b^{s-e^\dagger} \mod n \), i.e.,
  \((c', e, s^* = s-e^\dagger)\) 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 \( c' \)
- \( \text{PK}((\varepsilon, \mu_1, \sigma) : \ d/a_2^{m_2} := c'^\varepsilon a_1^{\mu_1} b^\sigma \land \mu \in \{0,1\}^\ell \land \varepsilon > 2^{\ell+1} ) \)
Password-based Security
Password are insecure, aren't they?

Username-password the most prominent form of user authentication

Passwords inherently insecure?

No! We're just using them incorrectly
The problem with passwords

Passwords are symmetric secrets → need protection on server & user

Password (hashes) useless against offline attacks
- Human-memorizable passwords are inherently weak
- NIST: 16-character passwords have 30 bits of entropy ≈ 1 billion possibilities
- Rig of 25 GPUs tests 350 billion possibilities / second, so ≈ 3ms for 16 chars
- 60% of LinkedIn passwords cracked within 24h

More expensive hash functions provide very little help only
- increases verification time as well
- does not work for short passwords such as pins etc

Single-server solutions inherently vulnerable to offline attacks
- Server / administrator / hacker can always guess & test
The solution: distributed password verification

Setup: Open account w/ password p

![Diagram showing distributed password verification process]

- User inputs password p
- Password is split into p1 and p2
- p = p1 ⋅ p2
- Each part is sent to different servers for verification

© 2016 IBM Corporation
The solution: distributed password verification

Login to account with password $p'$

- no server alone can test password
- passwords safe as long as not all servers are hacked
  - off-line attacks no longer possible
  - on-line attacks can be throttled
- pro-active re-sharing possible
- First server
  - web-server $\rightarrow$ replaces hash-data files
  - user's computer $\rightarrow$ secure against loss or theft of user device
How it works in a nutshell

- Servers share encryption secret key $x_1$ and $x_2$ for PK $X$ of a homomorphic scheme
- At setup: user encrypts $p$ under $X$: $E = Enc_X(p)$
- Password verification: check for encryption of 1

- Servers do not learn anything
  - $-1$ if passwords match, random number otherwise
- User could even be talking to the wrong servers...

\[
E' = (\text{Enc}_X(1/p') \diamond E)^r
= \text{Enc}_X((p/p')^r)
\]
One of the servers could be your smart phone, laptop, …

- Get key share from if password check succeeded
- Decrypt all your files on phone (or stored in the cloud, etc)
One of the servers could be your smart phone, laptop, …

- Get key share from if password check succeeded
- Decrypt all your files on phone (or stored in the cloud, etc)
Further Research Needed!

- **Securing the infrastructure & IoT**
  - “ad-hoc” establishment of secure authentication and communication
  - audit-ability & privacy (where is my information, crime traces)
  - security services, e.g., better CA, oblivious TTPs, anon. routing, ...

- **Usability**
  - HCI
  - Infrastructure (setup, use, changes by end users)

- **Provably secure protocols**
  - Properly modeling protocols (UC, realistic attacks models, ...)
  - Verifiable security proofs
  - Retaining efficiency
Further Research Needed!

- **Quantum Computers**
  - Lots of new crypto needed still
  - Build apps algorithm agnostic

- **Towards a secure information society**
  - Society gets shaped by quickly changing technology
  - Consequences are hard to grasp yet
  - We must inform and engage in a dialog
Conclusion

Let engage in some rocket science!

- Much of the needed technology exists
- … need to use them & build apps “for the moon”
- … and make apps usable & secure for end users

Thank you!

Joint work w/ Maria Dubovitskaya, Anja Lehmann, Anna Lysyanskaya, Gregory Neven, and many many more.

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