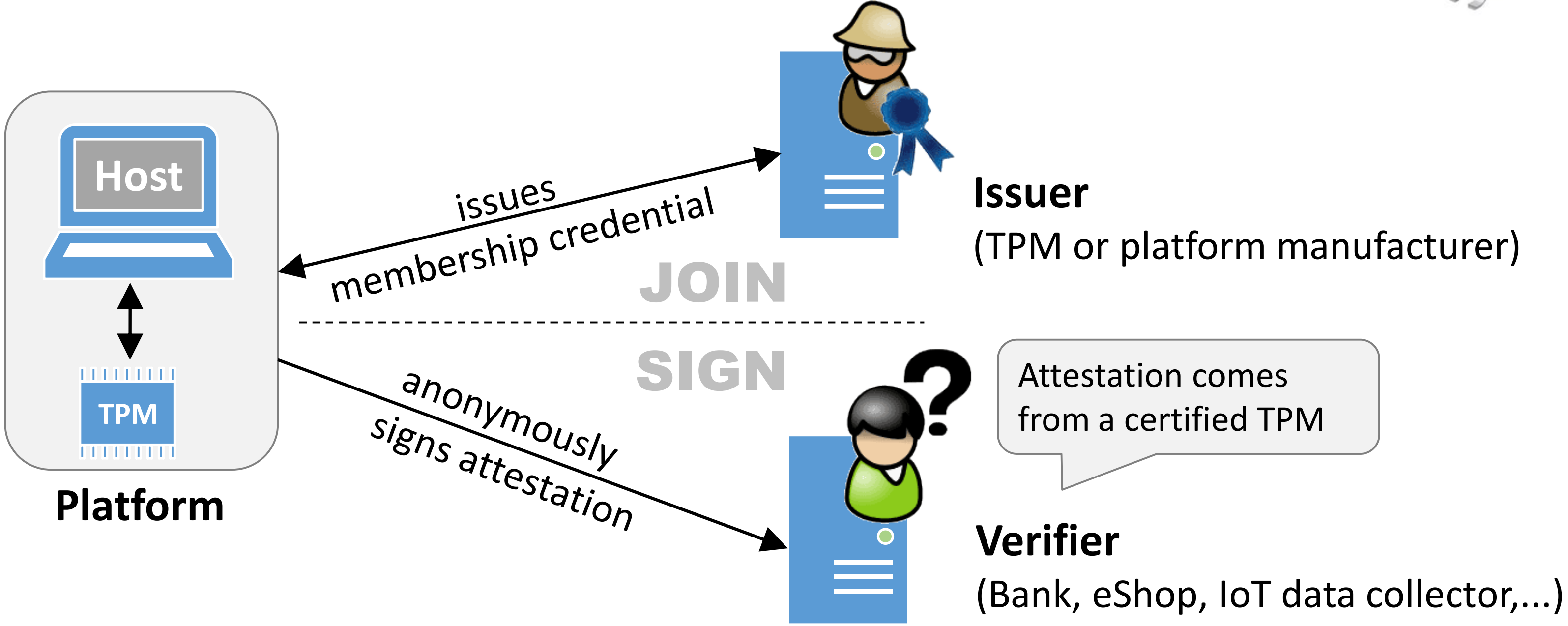
Direct Anonymous Attestation & TPM2.0

Getting Provably Secure Crypto into the Real-World

Anja Lehmann
IBM Research – Zurich

Direct Anonymous Attestation & TPM

- Trusted Platform Module (TPM)
  - Secure crypto processor: creates, stores, uses cryptographic keys
  - Makes remote attestations of host status



Platform

issues membership credential

JOIN

SIGN

anonymously signs attestation

Issuer
(TPM or platform manufacturer)

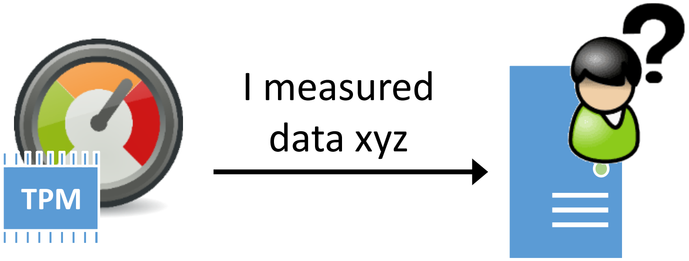
Attestation comes from a certified TPM

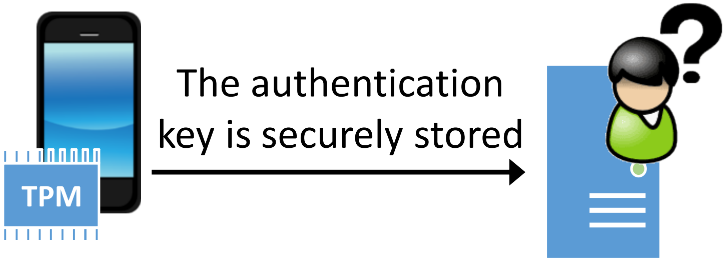
Verifier
(Bank, eShop, IoT data collector,...)

- Standard certificates would make all attestations linkable and reveal TPM‘s ID
- Direct Anonymous Attestation (DAA)
  - Security properties: unforgeability, anonymity & unlinkability, non-frameability

DAA – A Brief History

- First DAA protocol by Brickell, Camenisch, Chen [BCC04]
  - RSA-based
  - Standardized in TPM1.2 (2004) & ISO/IEC 20008-2
- Revised TPM2.0 (2014)
  - Elliptic curve & pairing based
  - Flexible API to support different protocols
  - TPM part & protocols ISO standardized
    - ISO/IEC 20008-2
    - ISO/IEC 11889
- Over 500 million TPMs sold
- Today: Interest in TPM revived
  - Security of mobile and IoT devices
  - FIDO authentication
  - SGX & EPID





Roadmap for Provable-Secure Real-World Crypto

- What is needed to make DAA a provably secure real-world protocol?

1. Security Model
2. Provably Secure Cryptographic Protocol (secure according to 1.)
3. Secure Implementation (of 2.)

...lets see where we are now, 12 years after DAA was invented

Security Models

Simulation-based Definitions

- Brickell, Camenisch, Chen [BCC04]
  - Does not output signatures
  - Prohibits working with signatures in practice
- Chen, Morissey, Smart [CMS08a]
  - Output signatures = random values
  - Not realizable by *any* construction
- Camenisch, Drijvers, Lehmann [CDL16a]
  - Security model in UC Framework
  - TPM and host separate parties
  - Signatures modeled as concrete values – for random TPM keys

Game-based Definitions

- Brickell, Chen, Li [BCL09]
  - Trivially forgeable scheme can be proven secure
  - No property for non-frameability
- Chen [C10], Brickell, Li [BL10]
  - Extends [BCL09] with non-frameability and signature-based revocation, EPID
  - Same unforgeability flaw as [BCL09]
- Bernhard et al. [BFG+13]
  - Discuss flaws in all previous models
  - Extensive set of definitions for all expected properties
  - But for “pre-DAA”, where TPM + host are one party → does not cover honest TPM in corrupt host

All existing security definitions had issues, some of them severe, allowing for insecure schemes!

Roadmap for Provable-Secure Real-World Crypto

- What is needed to make DAA a provably secure real-world protocol?

1. Security Model ✓

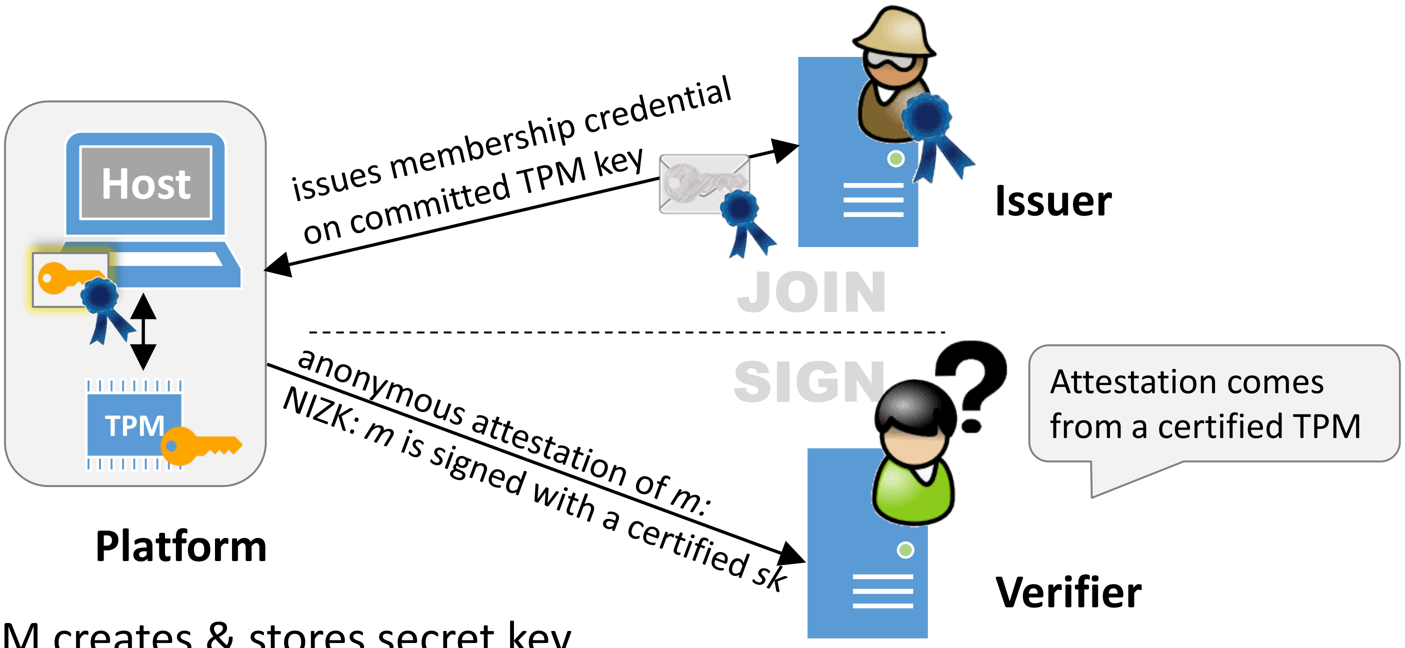
↓

2. Provably Secure Cryptographic Protocol (secure according to 1.)

↓

3. Secure Implementation (of 2.)

DAA Protocols – Common Approach

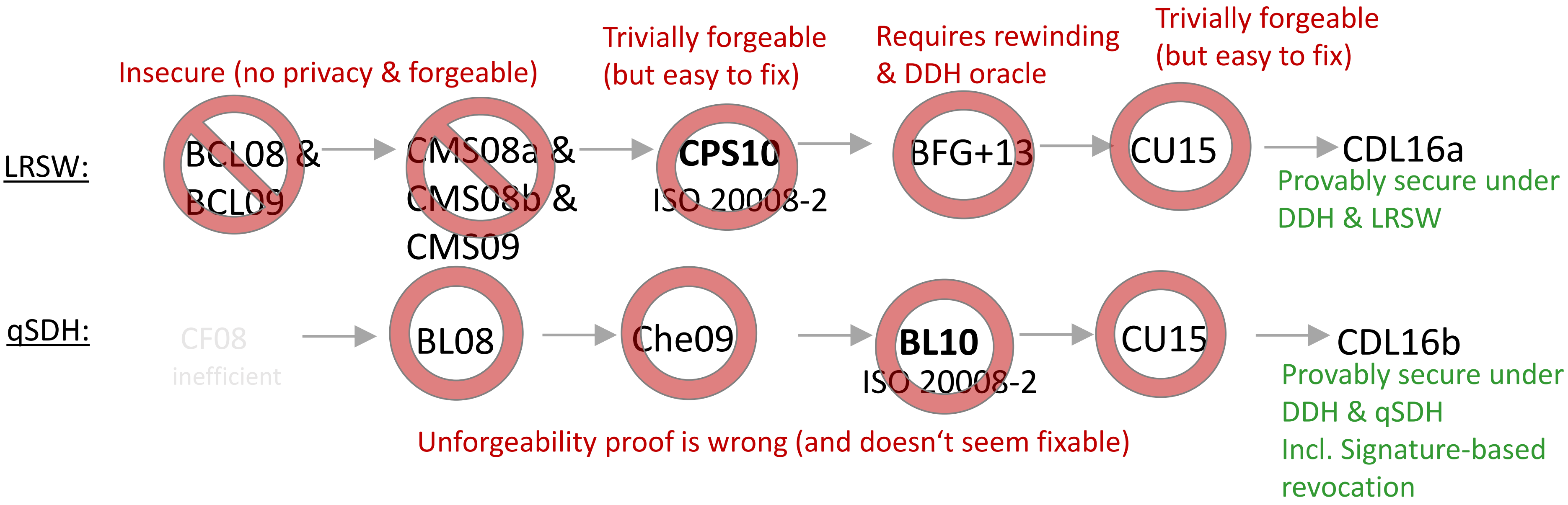

Host
TPM
Platform
issues membership credential
on committed TPM key
Issuer
JOIN
SIGN
anonymous attestation of *m*:
NIZK: *m* is signed with a certified *sk*
Attestation comes from a certified TPM
Verifier

TPM creates & stores secret key
Host stores membership credential

- DAA protocols mainly differ on how the membership credential & NIZK is computed
- First protocol [BCC04] based on RSA, standardized in TPM1.2 (very slow)
- Subsequent DAA protocols & TPM2.0 based on elliptic curves and pairings

DAA Protocols for TPM2.0

- TPM2.0 offers generic APIs to support various schemes,
  e.g., DAA based on LRSW (CL-signature) & qSDH (BBS+ signature)



LRSW:

Insecure (no privacy & forgeable)

BCL08 & BCL09 → CMS08a & CMS08b & CMS09

Trivially forgeable (but easy to fix)

→ **CPS10** ISO 20008-2

Requires rewinding & DDH oracle

→ BFG+13

Trivially forgeable (but easy to fix)

→ CU15 → CDL16a

Provably secure under DDH & LRSW

qSDH:

CF08 inefficient → BL08 → Che09 → **BL10** ISO 20008-2 → CU15 → CDL16b

Unforgeability proof is wrong (and doesn‘t seem fixable)

Provably secure under DDH & qSDH Incl. Signature-based revocation

- All existing schemes are either insecure, or cannot be proven secure
  – (1, 1, 1, 1) is a valid credential on *any* key in [CPS10] – ISO 20008 standardized!

- Revised provably secure protocols [CDL16a, CDL16b]
  – as efficient as existing schemes – mainly details had to be fixed

Roadmap for Provable-Secure Real-World Crypto

- What is needed to make DAA a provably secure real-world protocol?

1. Security Model ✓

↓

2. Provably Secure Cryptographic Protocol (secure according to 1.) ✓

↓

3. Secure Implementation (of 2.)

Efficient protocol, lightweight part for TPM …. ~~Done!~~

Our real-world:	TPM = lightweight device
Real real-world:	TPM accessible via few, limited APIs

TPM2.0 Interfaces (High-Level Idea)

TPM

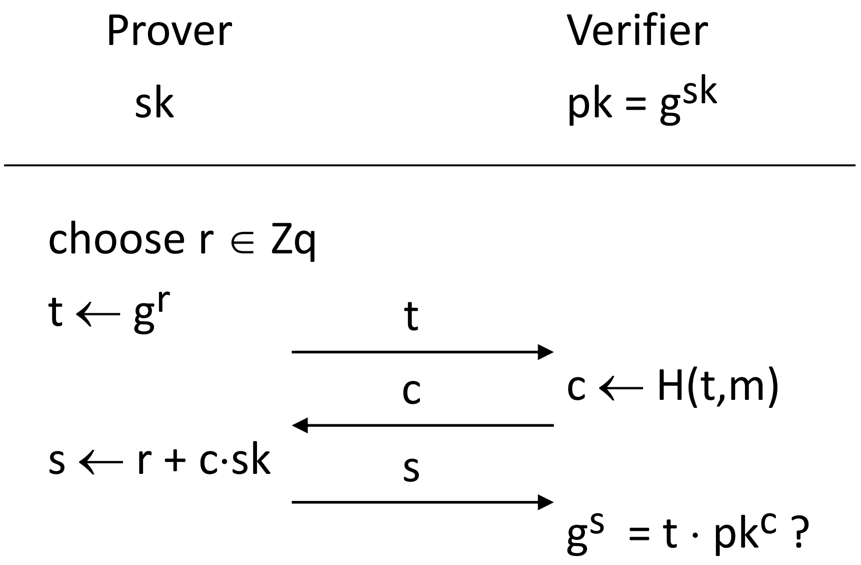
```
TPM.Create()
    draw $sk \in Zq$, store sk
    output $pk \leftarrow g^{sk}$

TPM.Hash(t,m)
    output $c \leftarrow H(t,m)$

TPM.Commit(P)
    choose $r \in Zq$ , store (ctr, r)
    $t \leftarrow P^{r}$
    output (ctr, t)

TPM.Sign(c, ctr)
    get (ctr, r)
    output $s \leftarrow r + c \cdot sk$
```

(Signature) Proof-of-Knowledge:



Prover

sk

Verifier

$pk = g^{sk}$

choose $r \in Zq$

$t \leftarrow g^{r}$ $\xrightarrow{t}$

$\xleftarrow{c}$ $c \leftarrow H(t,m)$

$s \leftarrow r + c \cdot sk$ $\xrightarrow{s}$

$g^{s} = t \cdot pk^{c}$?

- Both revised protocols are not compatible with current TPM2.0 interfaces
- Protocols designed to avoid a static Diffie-Hellman oracle – but TPM2.0 is one

TPM2.0 Interfaces | Static Diffie-Hellman Oracle

TPM

```
TPM.Create()
  draw $sk \in Zq$, store sk
  output $pk \leftarrow g^{sk}$

TPM.Hash(t,m)
  output $c \leftarrow H(t,m)$

TPM.Commit(P)
  choose $r \in Zq$ , store (ctr, r)
  $t \leftarrow P^{r}$
  output (ctr, t)

TPM.Sign(c, ctr)
  get (ctr, r)
  output $s \leftarrow r + c \cdot sk$
```

TPM2.0 interfaces provide static Diffie-Hellman Oracle

- DH oracle via Commit, Hash & Sign query:

$$P^{sk} \leftarrow (P^{s} / t)^{1/c}$$

For arbitrary P chosen by (corrupt) host

- Get TPM to compute $g^{sk}, g^{sk^2}, g^{sk^3} \cdots g^{sk^n}$
- Static DH oracle significantly reduces security level,

  e.g., 256bit BN curve: 128bit security reduced to 85bit

TPM interfaces should be revised to remove the static DH oracle!

TPM & DAA without a Static DH oracle | First Proposal

TPM

```
TPM.Create()
    draw sk $\in$ Zq, store sk
    output pk $\leftarrow g^{sk}$

TPM.Hash(t,m)
    output c $\leftarrow$ H(t,m)

TPM.Bind(P, K, $\pi$)
    verify that $\pi$ is valid
    store P as „cleared point“

TPM.Commit(P)
    abort if P is not a cleared
    choose r $\in$ Zq , store (ctr, r)
    t $\leftarrow P^r$
    output (ctr, t)

TPM.Sign(c, ctr)
    get (ctr, r)
    output s $\leftarrow$ r + c $\cdot$sk
```

```
TPM.Commit2(bsn)
    choose r $\in$ Zq , store (ctr, r)
    P $\leftarrow$ H(bsn), t $\leftarrow P^r$
    output (ctr, t)
```

Cleared Generators (Xi et al. [XYZF14])

- Generator has form $P = g^y$
- Issuer knows y ... And therefore $K = P^{sk} = pk^y$

  $\pi \leftarrow \mathrm{SPK}\{(y): P = g^y \text{ and } K = pk^y\}$

Random (Hashed) Generators

- P is chosen at random, input to attestation
- Use P ← H(bsn) for random bsn

OK, but only if it can be used for qSDH & LRSW DAA!

TPM & DAA Revisited | Real-World Compatible Proposal

TPM

```
TPM.Create()
    draw $sk \in Zq$, store sk
    output $pk \leftarrow g^{sk}$

TPM.Hash(t,m)
    output $c \leftarrow H(t,m)$

TPM.Commit(bsn)
    choose $r \in Zq$ , store (ctr, r)
    $P \leftarrow H(bsn)$,  $t \leftarrow P^r$
    output (ctr, t)

TPM.Sign(c, ctr)
    get (ctr, r)
    output $s \leftarrow r + c \cdot sk$
```

Revised TPM2.0 interfaces w/o static DH ✓

Re-revised provably secure LRSW/qSDH-DAA ✓

Are we done now?

Are the TPM-based contributions unforgeable & anonymous ?

- Chen, Li [CL13]
  - Proof that TPM2.0 generated SPKs are unforgeable
- Xi et al. [XYZF14]
  - Proof by [CL13] is wrong
  - Unforgeability cannot be proven

TPM & DAA Revisited | Real-World Compatible Proposal

TPM

```
TPM.Create()
    draw sk $\in$ Zq, store sk
    output pk $\leftarrow g^{sk}$

TPM.Hash(t,m)
    output c $\leftarrow$ H(t,m)

TPM.Commit(bsn)
    choose r $\in$ Zq , store (ctr, r)
    P $\leftarrow$ H(bsn),  t $\leftarrow P^r$
    output (ctr, t)

TPM.Sign(c, ctr)
    get (ctr, r)
    random n, c' $\leftarrow$ H(n, c)
    output n, s $\leftarrow$ r + c' $\cdot$sk
```

Revised TPM2.0 interfaces w/o static DH ✓

Re-revised provably secure LRSW/qSDH-DAA ✓

Are we done now?

Are the TPM-based contributions unforgeable & anonymous ?

- Chen, Li [CL13]
  - Proof that TPM2.0 generated SPKs are unforgeable
- Xi et al. [XYZF14]
  - Proof by [CL13] is wrong
  - Unforgeability cannot be proven
  - Simple Fix: add nonce and hash

TPM & DAA Revisited | Real-World Compatible Proposal

TPM

```
TPM.Create()
    draw $sk \in Z_q$, store sk
    output $pk \leftarrow g^{sk}$

TPM.Hash(t,m)
    output $c \leftarrow H(t,m)$

TPM.Commit(bsn)
    choose $r \in Z_q$, store (ctr, r)
    $P \leftarrow H(bsn)$,  $t \leftarrow P^r$
    output (ctr, t)

TPM.Sign(c, ctr)
    get (ctr, r)
    random n, $c' \leftarrow H(n, c)$
    output n, $s \leftarrow r + c' \cdot sk$
```

Revised TPM2.0 interfaces w/o static DH ✓

Re-revised provably secure LRSW/qSDH-DAA ✓

Are we done now?

The TPM-based contributions are unforgeable & anonymous ✓

- Fix by Xi et al. introduces subliminal channel!

TPM & DAA Revisited | Real-World Compatible Proposal

TPM

```
TPM.Create()
  draw $sk \in Zq$, store sk
  output $pk \leftarrow g^{sk}$

TPM.Hash(t,m)
  output $c \leftarrow H(t,m)$

TPM.Commit(bsn)
  random nT,  $hT \leftarrow H(nT)$
  choose $r \in Zq$ , store (ctr, r, nT)
  $P \leftarrow H(bsn)$,  $t \leftarrow P^r$
  output (ctr, t, hT)

TPM.Sign(c, ctr, nH)
  get (ctr, r, nT)
  $c' \leftarrow H(nH \oplus nT , c)$
  output nT, $s \leftarrow r + c' \cdot sk$
```

Revised TPM2.0 interfaces w/o static DH ✓

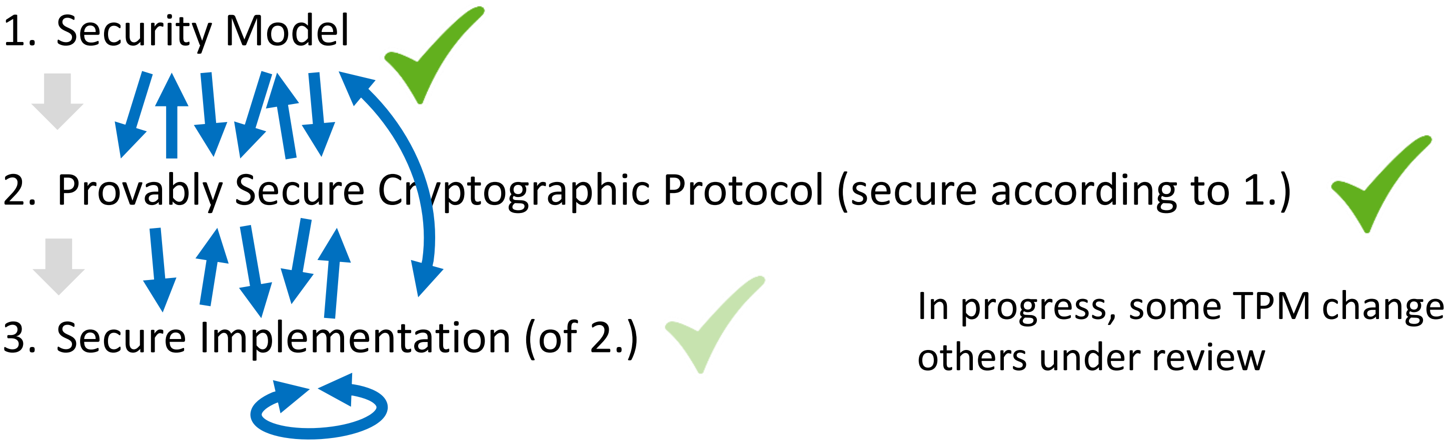
Re-revised provably secure LRSW/qSDH-DAA ✓

Are we done now?

The TPM-based contributions are unforgeable & anonymous ✓

- Fix by Xi et al. introduces subliminal channel!
- New fix: use nonce jointly computed by host and TPM

Roadmap for Provable-Secure Real-World Crypto



1. Security Model ✓
2. Provably Secure Cryptographic Protocol (secure according to 1.) ✓
3. Secure Implementation (of 2.) ✓

In progress, some TPM changes accepted, others under review

Next Steps:

- Continue work with TCG on revision of TPM2.0 APIs
- Working to get flawed ISO standards fixed
- Working with Intel on revision of EPID spec
- FIDO key attestation spec using DAA

Conclusions:

- Provably secure crypto and real-world should be compatible
- Ideally, provable security from the beginning – a number of standards have issues!
- It often takes far longer than one would expect & still not done

Thanks!

Based on joint work with Jan Camenisch, Liqun Chen, Manu Drijvers, David Novick, Rainer Urian

Questions?

ia.cr/2015/1246
ia.cr/2016/663
anj@zurich.ibm.com

References

[BCC04] Brickell, Camenisch, Chen. Direct anonymous attestation. ACM CCS 04

[BCL08] Brickell, Chen, Li. A new direct anonymous attestation scheme from bilinear maps. Trust 2008

[BCL09] Brickell, Chen, Li. Simplied security notions of DAA and a concrete scheme from pairings. Int. J. Inf. Sec., 2009.

[BFG+13] Bernhard, Fuchsbauer, Ghada, Smart, Warinschi. Anonymous attestation with user-controlled linkability. Int. J. Inf. Sec., 2013.

[BL07] Brickell, Li. Enhanced privacy id: a direct anonymous attestation scheme with enhanced revocation capabilities. WPES 2007

[BL10] Brickell, Li. A pairing-based DAA scheme further reducing TPM resources. Trust 2010

[BL11] Brickell, Li. Enhanced privacy ID from bilinear pairing for hardware authentication and attestation. IJIPSI, 1(1):3{33, 2011.

[CDL16a] Camenisch, Drijvers, Lehmann. Universally composable direct anonymous attestation. PKC 2016

[CDL16b] Camenisch, Drijvers, Lehmann. Anonymous attestation using the strong diffie hellman assumption revisited. Trust 2016

[C10] Chen. A DAA scheme requiring less TPM resources. Inscrypt 2010

[CF08] X. Chen, Feng. Direct anonymous attestation for next generation TPM. JCP, 3(12):43{50, 2008.

[CMS08a] Chen, Morrissey, Smart. Pairings in trusted computing (invited talk). PAIRING 2008.

[CMS08b] Chen, Morrissey, Smart. On proofs of security for DAA schemes. Provable Security 2008.

[CMS09] Chen, Morrissey, Smart. DAA: Fixing the pairing based protocols. ePrint Archive, Report 2009/198.

[CPS10] Chen, Page, Smart. On the design and implementation of an efficient DAA scheme. CARDIS 2010