Cryptography with Tamperable and Leaky Memory

Yael Tauman Kalai   Bhavana Kanukurthi   Amit Sahai

MSR   UCLA   UCLA

Leakage Resilient Cryptography

[Rivest1997, Boyko1999, Canetti-Dodis-Halevi-Kushilevitz-Sahai2000, Ishai-Sahai-Wagner2003, Micali-Reyzin2004, Ishai-Prabhakaran-Sahai-Wagner2006, Dziembowski-Pietrzak2008, Pietrzak2009 , Akavia-Goldwasser-Vaikuntanathan2009, Dodis-Kalai-Lovett2009, Naor-Segev2009, Katz-Vaikuntanathan2009, Alwen-Dodis-Wichs2009, Alwen-Dodis-Naor-Segev-Walfish-Wichs2009, Faust-Kiltz-Pietrzak-Rothblum2009, Faust-Rabin-Reyzin-Tromer-Vaikuntanathan2010, Dodis-Goldwasser-Kalai-Peikert-Vaikuntanathan2010, Goldwasser-Kalai-Peikert-Vaikuntanathan2010, Juma-Vahlis2010, Goldwasswer-Rothblum2010, Canetti-Kalai-Mayank-Wichs2010, Dodis-Haralambiev-LopezAlt-Wichs2010, Brakerski-Kalai-Katz-Vaikuntanathan2010, Boyle-Segev-Wichs2010, Dodis-Pietrzak2010, Braverman-Hassidim-K2010, Lewko-Waters2010, Lewko-Rouselakis-Waters2011, Lewko-Lewko-Waters2011]

We know how to build cryptographic scheme that are secure against continual leakage!

[Dodis-Haralambiev-LopezAlt-Wichs2010, Brakerski-Kalai-Katz-Vaikuntanathan2010]

BUT physicals attacks aren’t restricted to leakage attacks;
they also tamper with the memory!

[Considered for e.g., in Biham and Shamir Crypto ’97; Boneh-DeMillo-Lipton Eurocrypt ‘97, Kocher-Jaffe-Jun Crypto ’99, Govindavajhala and Appel IEEE Symposium on S&P ’03]

Prior Work: Tamper Resilient Cryptography

- [Gennaro, Lysysanskaya, Malkin, Micali, Rabin TCC ’04]:
  - Achieve strong tamper–proof security but rely on some *non–tamperable (user–specific)* memory.
- [Ishai, Prabhakaran, Sahai, Wagner Eurocrypt ’06]:
  - Considered tampering applied to all parts of computation.
  - But consider only tampering functions that set/reset bits.
- [Bellare, Kohno Eurocrypt ’03], [Dziembowski, Pietrzak, Wichs, ICS ‘10], [Applebaum, Harnik, Ishai ICS ‘11]
  - Limited tampering to memory.

Our Goals

Build leakage and tamper resilient that always satisfy the following conditions:

- All user–modifiable memory is *tamperable* and *leaky*; (in particular, the public key stored on device is also tamperable).
  - Note that public/private keys must be part of user-modifiable memory, since they are unique to each user.
- Allow for arbitrary tampering and leakage.

We achieve this!

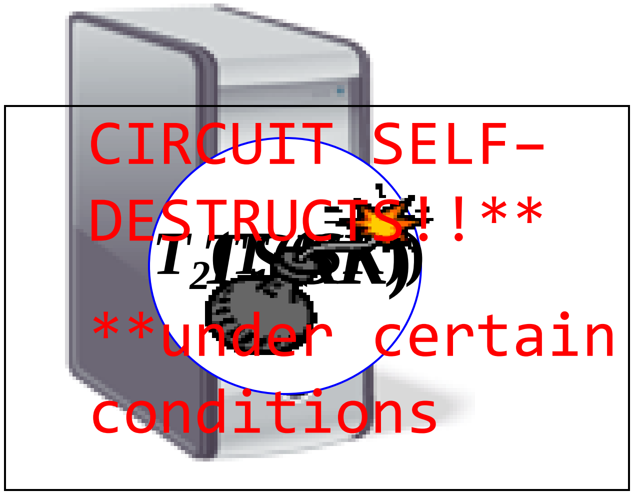
- Assume *non–tamperable public parameters* (CRS).
- Rely on a source of true local randomness. (Necessary for our setting: Lysysanskaya, Liu SCN '10)

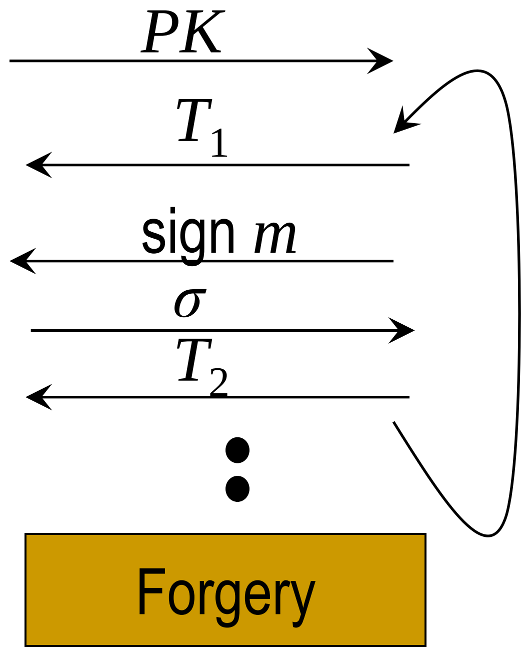
Our Results (Informally)

Result 1: We present a general transformation that converts any scheme resilient to bounded leakage into one that is also resilient to continual tampering. (Instantiable using FHE + NIZKs.)

Result 2: We construct encryption and signature schemes resilient to continual leakage and tampering, based on linear assumptions over bilinear groups.

Signature Scheme in the Continual Tampering Model





Success: if *forgery* verifies wrt *original PK*

Easy to see: This is impossible to achieve!

Problem: Adversary can tamper with *sk* bit-by-bit and use her signature queries to learn the entire secret key!

FIX: Need to assume that the circuit self–destructs!

Building Block: NIZK Proofs of Knowledge

Common Reference String (CRS)

Goal: Prove statement X in$L$

Prover

witness (w)

Verifier

$$\pi = P(CRS, x, w) \longrightarrow$$

We require our NIZK proof system to have some additional properties:

- Simulation soundness: Hard to prove false statements *even after seeing* simulated proofs of false statements.
- Proof of Knowledge: If adversary outputs a valid proof, then the simulator can extract a witness out of it.
- SHORT proof: Length of π *should depend polynomially on* $|w|$.

Our General Transformation

$\mathcal{S}$ = (Gen, Sig, Ver) is a leakage resilient signature scheme

- with $sk \leftarrow \{0,1\}^n$ and pk efficiently generated from sk

$\mathcal{S}'$ = (Gen', Sig', Ver') is the tamper resilient scheme we build from $\mathcal{S}$.

- Gen':
  - Sets sk: PRG(r)

    “short” simulation sound

  - $sk' := (sk, \pi)$ (where π: NIZK proof of pseudo–randomness)

    proof of knowledge

- Sig'$_{sk'}(m)$:
  - First verifies $sk' := (sk, \pi)$ is valid (self–destructs otherwise).
  - Returns Sig$_{sk}(m)$

Informal Theorem: If $\mathcal{S}$ is resilient to $|r| + |\pi|$ bits of leakage, then $\mathcal{S}$' is resilient to continual tampering;

(where r: PRG seed;
π: NIZK proof of pseudo–randomness).

Intuition behind Security

Leakage Challenger C	$\xrightarrow{pk}$	Leakage Adversary $\mathcal{B}$	$\xrightarrow{pk,\ crs}$	Tampering Adversary $\mathcal{A}$

(sk, pk)
where $sk \leftarrow \{0,1\}^n$

$(crs$, with trapdoor $\mu)$

$\xleftarrow{Sign\ m}$

$\xrightarrow{\sigma}$

$\xleftarrow{\text{Leakage } \mathcal{L}}$ (to be applied on (sk, pk))

Sig and Sig' are equivalent until the secret key has been tampered with!

$\xleftarrow{Sign\ m}$

$\xrightarrow{\sigma}$

$\xleftarrow{\text{Tamper } \mathcal{T}}$ (to be applied on $((sk, \pi), pk)$)

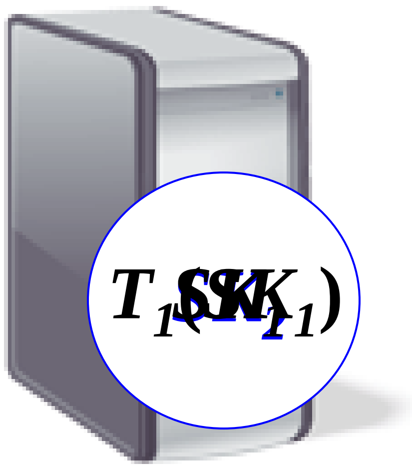
- $\mathcal{L}(sk, pk)$:
  - Computes π := SimProof that "sk is pseudo-random".
  - Sets $(sk^\star, \pi^\star, pk^\star) := \mathcal{T}(sk, \pi, pk)$.
  - If proof is valid, then $sk^\star = PRG(r^\star)$, so can extract $r^\star$

Extracts $r^\star$ from $\pi(sk^\star, m, pk)$

$\xrightarrow{r^\star,\ \pi^\star,}$

With $(r^\star, \pi^\star)$, $\mathcal{B}$ has the current secret state (i.e., $sk^\star, \pi^\star$) entirely; so she can simulate rest of $\mathcal{A}$'s queries on her own.

Signature Scheme in Continual Tampering and Memory Leakage Model



SK_2 = Update($T_1\,(SK_1)\,)$

PK

L_1 L_2

$L_1(SK_1)$ $L_2(SK_1)$

T_1

sign m

σ

UPDATE

More leakage, tampering & signature queries (in any order)

:

Forgery

Bounded amount of leakage

Starting Point for our work: Continual Memory Leakage Scheme of BKKV

NOTE: amount of leakage that the adversary gets in the entire lifetime of the secret key is not bounded

Main Challenge:
How do you secure updates with tampered secret keys?

Success: if *forgery* verifies wrt *PK*

Our Continual Tamper and Leakage Resilient Scheme

(NOTE: PP is non-tamperable; but not user specific)

Public Parameters: $(g^{\bar{a}}, g^{\bar{b}})$ such that $\bar{a} \cdot \bar{b} = 0$ where $\bar{a}, \bar{b} \leftarrow \mathbb{Z}_p^{\ell}$

Secret Key: $g^{\bar{s}}$ where $\bar{s} \leftarrow \mathbb{Z}_p^{\ell}$

Public Key: $e(g^{\bar{a}}, g^{\bar{s}}) = e(g, g)^{\bar{a} \cdot \bar{s}}$

Update: $g^{\bar{s}} \xrightarrow{\text{update}} g^{\bar{s}+\alpha_1 \bar{b}} \xrightarrow{\text{update}} g^{\bar{s}+\alpha_1 \bar{b}+\alpha_2 \bar{b}}$

See paper for details!

Conclusion

- This talk: Presented a generic transformation that converts bounded leakage resilience to (leakage) and tamper resilience.

- Presented the first number-theoretic construction of cryptographic schemes simultaneously resilient to continual leakage and tampering.

Thank you!!!