Public-Key Cryptosystems Resilient to Key Leakage

Moni Naor

Gil Segev

Weizmann Institute of Science

Israel

Foundations of Cryptography

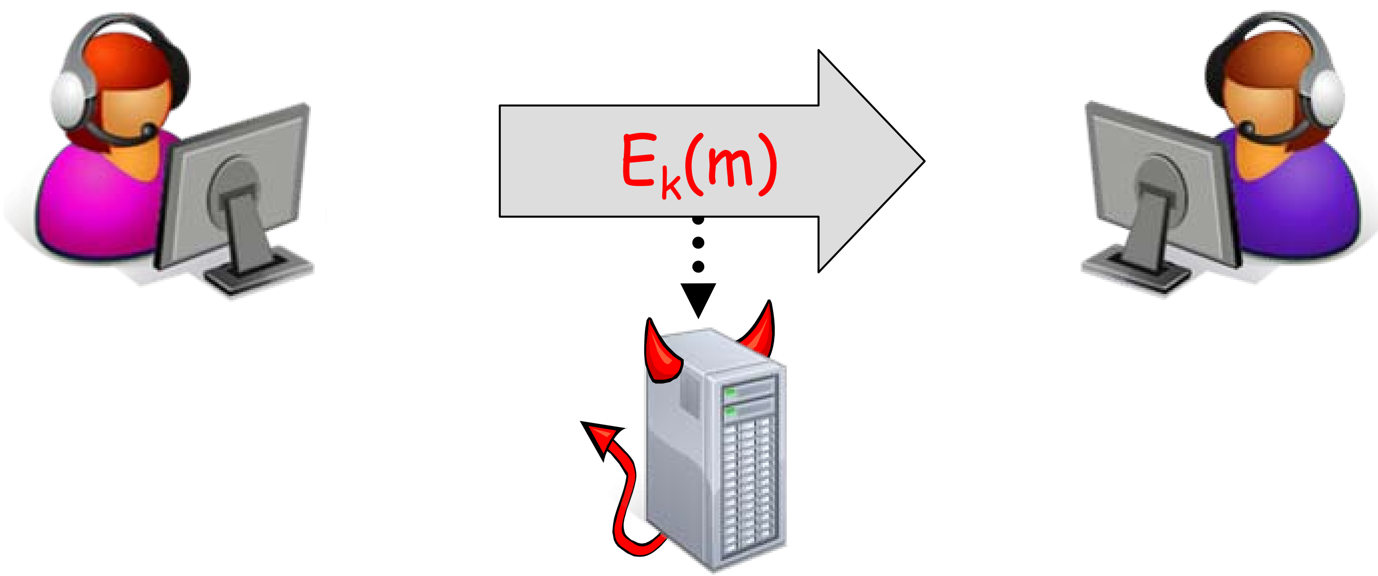
Rigorous analysis of the security of cryptographic schemes

Adversarial model

- Computational capabilities
- Access to the system

Notion of security

- What does it mean to break the system?



Foundations of Cryptography

Rigorous analysis of the security of cryptographic schemes

Adversarial model
- Computational capabilities
- Access to the system

Notion of security
- What does it mean to break the system?

- Notions of security significantly evolved 
- Adversarial access analyzed in the "standard model"...

Adversarial Models

STANDARD MODEL:

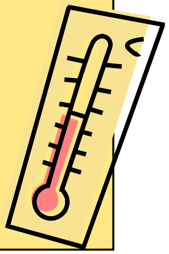
- Abstract computation
  - Interactive Turing machines
  - Private memory & randomness
- Well-defined adversarial access
- Can model powerful attacks
  - CPA\CCA, composition, key cycles,...

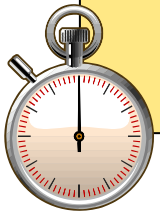
REAL LIFE:

- Physical implementations leak information
- Side-channel attacks
  - Timing attacks [Kocher 96]
  - Fault detection [BDL 97, BS 97]
  - Power analysis [KJJ 99]
  - Cache attacks [OST 05]
  - Memory attacks [HSHCPCFAF 08]

SIDE CHANNEL:

Any information not captured by the underlying model





Modeling Side Channels

- Canetti, Dodis, Halevi, Kushilevitz, and Sahai ’00
  Exposure-resilient functions: functions that “look” random even if several input bits are leaked

- Ishai, Prabhakaran, Sahai, and Wagner ’03 ’06
  Private circuit evaluation allowing several wires to leak

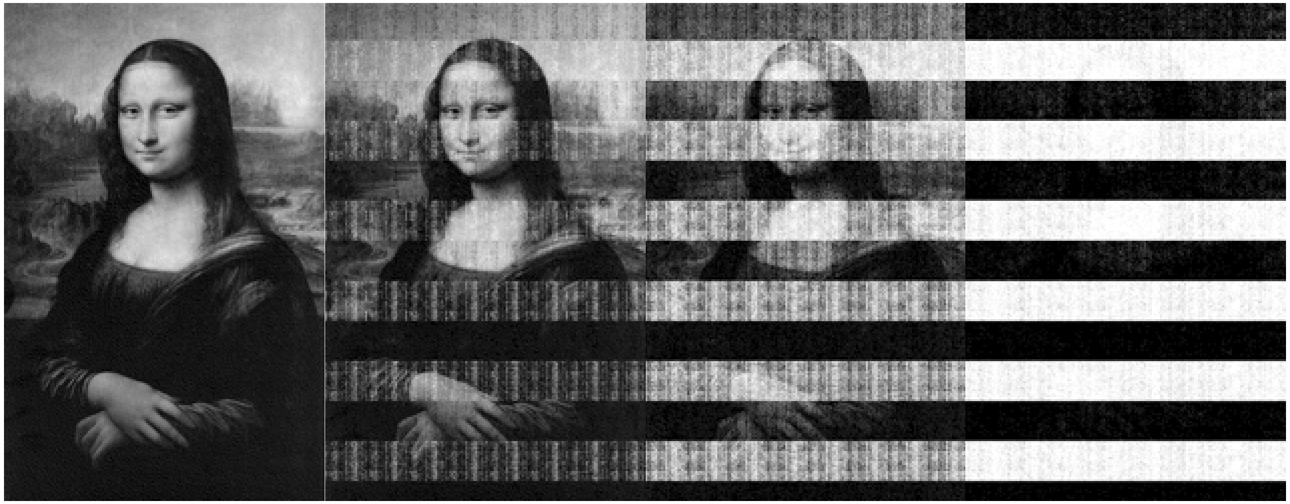
- Micali and Reyzin ’04
  Computation and only computation leaks information

- Dziembowski and Pietrzak ’08, Pietrzak ’09
  Leakage-resilient stream-ciphers
  - Computation and only computation leaks information
  - Low-bandwidth leakage

Memory Attacks [HSHCPCFAF 08]

Halderman, Schoen, Heninger, Clarkson, Paul, Calandrino, Feldman, Appelbaum and Felten

- Not only computation leaks information
- Memory retains its content after power is lost



5 seconds | 30 seconds | 60 seconds | 5 minutes

http://citp.princeton.edu/memory

Memory Attacks [HSHCPCFAF 08]

- Not only computation leaks information
- Memory retains its content after power is lost

Memory content can even last for several minutes

- Recover "noisy" keys
  - Cold boot attacks
  - Completely compromise popular disk encryption systems
  - Reconstruct DES, AES, and RSA keys

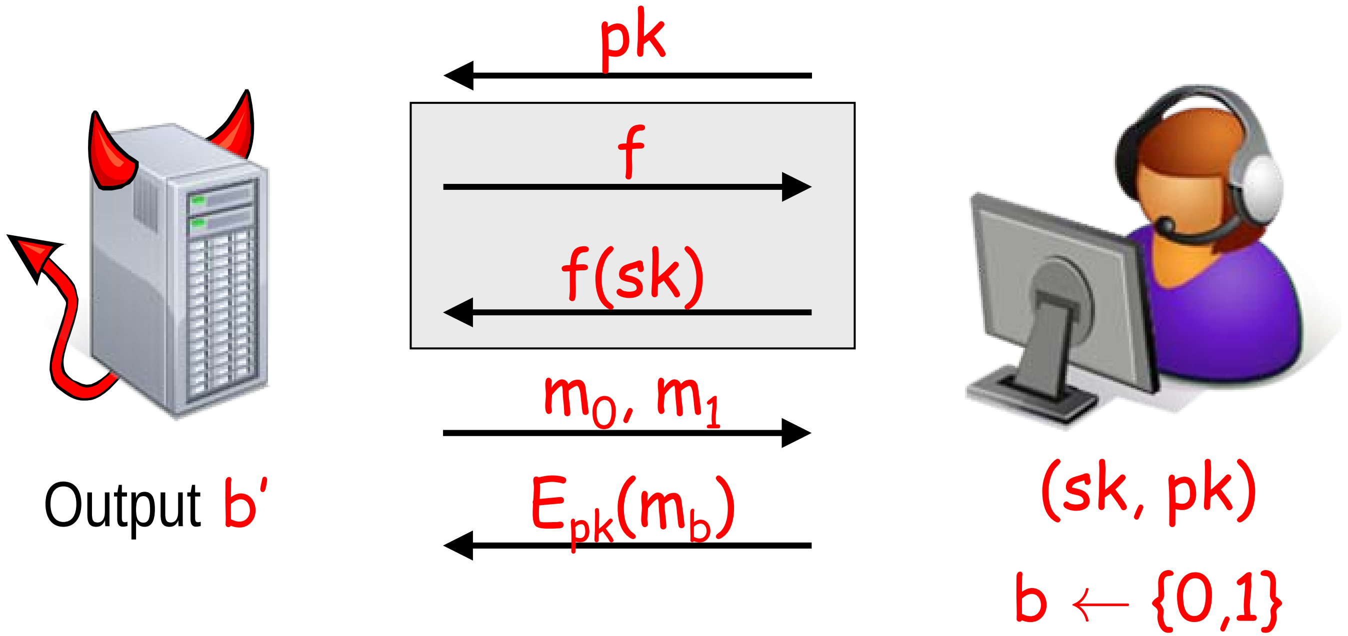
Extended and further analyzed by Heninger & Shacham 09

http://citp.princeton.edu/memory

Memory Attacks

Akavia, Goldwasser & Vaikuntanathan

Semantic security with key leakage [AGV 09]:
For any* leakage $f(sk)$ and for any m_0 and m_1 infeasible to distinguish $E_{pk}(m_0)$ and $E_{pk}(m_1)$



- Clearly, cannot allow $f(sk)$ that easily reveals sk
- For now $f : SK \rightarrow \{0,1\}^{\lambda}$ for $\lambda < |sk|$

[AGV 09]: Regev's lattice-based scheme is resilient to such leakage

Our Results

- A generic construction secure against key leakage
  - Based on any Hash Proof System [CS 02]
  - Efficient instantiations
  - Various number-theoretic assumptions

- A new hash proof system
  - Resulting scheme resilient to leakage of $L - o(L)$ bits
  - Based on either DDH or d-Linear

- The [BHHO 08] circular-secure scheme
  - Fits into our generic approach
  - Resilient to leakage of $L - o(L)$ bits

Our Results

- Chosen-ciphertext security

Theoretical side

- A generic CPA-to-CCA transformation
- Leakage of $L - o(L)$ bits

Practical side

- Efficient variants of Cramer-Shoup
- CCA1: Leakage of $L/4$ bits
- CCA2: Leakage of $L/6$ bits

- Extensions of the [AGV 09] model
  - Noisy leakage
  - Leakage of intermediate values
  - Keys generated using a "weak" random source

} Satisfied by our schemes

Independently by Tauman Kalai & Vaikuntanathan: [BHHO 08] with hard-to-invert leakage and CPA-to-CCA

Outline of the Talk

- The generic construction by example
  - An efficient scheme with $\lambda \approx |sk|/2$
- Extensions of the model
- Conclusions & open problems

A Simple Scheme

- G - group of order p in which DDH is hard
- $Ext : G \times \{0,1\}^d \to \{0,1\}$ - strong extractor

Key generation

- Choose $g_1, g_2 \in G$ and $x_1, x_2 \in Z_p$
- Let $h = g_1^{x_1} g_2^{x_2}$
- Output $sk = (x_1, x_2)$ and $pk = (g_1, g_2, h)$

MAIN IDEA

- Redundancy: pk corresponds to many possible sk's
- $h=g_1^{x_1} g_2^{x_2}$ reveals only $\log(p)$ bits of information on $sk=(x_1,x_2)$
- Leakage of λ bits $\Rightarrow$ sk still has min-entropy $\log(p) - \lambda$

A Simple Scheme

- G - group of order p in which DDH is hard
- $\mathrm{Ext} : G \times \{0,1\}^d \to \{0,1\}$ - strong extractor

Key generation

- Choose $g_1, g_2 \in G$ and $x_1, x_2 \in Z_p$
- Let $h = g_1^{x_1} g_2^{x_2}$
- Output $sk = (x_1, x_2)$ and $pk = (g_1, g_2, h)$

$\mathrm{Enc}_{pk}(m)$

- Choose $r \in Z_p$ and a seed $s \in \{0,1\}^d$
- Output $(g_1^r, g_2^r, s, \mathrm{Ext}(h^r, s) \oplus m)$

$\mathrm{Dec}_{sk}(u_1, u_2, s, e)$

- Output $e \oplus \mathrm{Ext}(u_1^{x_1} u_2^{x_2}, s)$

Correctness: $u_1^{x_1} u_2^{x_2} = (g_1^{x_1} g_2^{x_2})^r = h^r$

Security of the Simple Scheme

Theorem: The scheme is resilient to any leakage of $\lambda \approx \log(p)$ bits

half the size of sk

Proof by reduction:

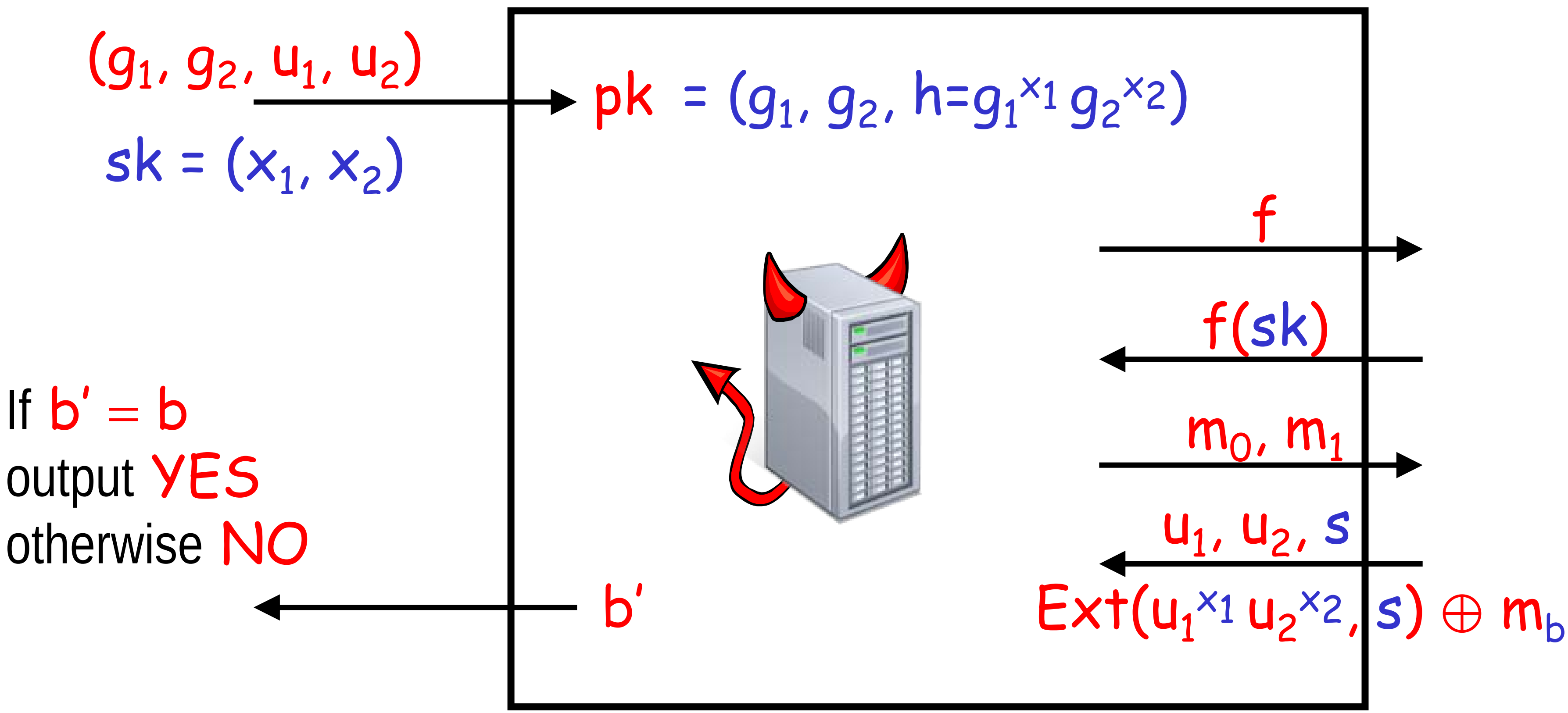
Adversary for the encryption scheme → Algorithm for DDH:

(g_1, g_2, g_1^r, g_2^r)

or

$(g_1, g_2, g_1^{r_1}, g_2^{r_2})$

The Reduction



Case 1: $u_1 = g_1^r$ & $u_2 = g_2^r$

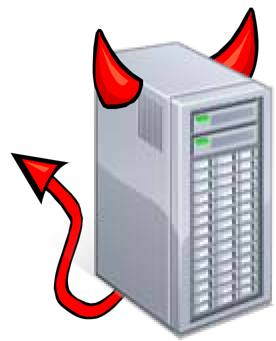
$u_1^{x_1} u_2^{x_2} = (g_1^{x_1} g_2^{x_2})^r = h^r$

- Simulation is identical to actual attack
- By assumption $Pr[b' = b] > 1/2 + 1/poly$

The Reduction

(g_1, g_2, u_1, u_2) → pk $= (g_1, g_2, h = g_1^{x_1} g_2^{x_2})$

sk $= (x_1, x_2)$



f →

← f(sk)

m_0, m_1 →

← u_1, u_2, s

$\mathrm{Ext}(u_1^{x_1} u_2^{x_2}, s) \oplus m_b$

← b′

If $b' = b$
output YES
otherwise NO

Case 2: $u_1 = g_1^{r_1}$ & $u_2 = g_2^{r_2}$

- Challenge independent of b
- $\Pr[b' = b] = 1/2$

$u_1^{x_1} u_2^{x_2}$ is uniform in G

λ bits of leakage $\Rightarrow$

$H_\infty(u_1^{x_1} u_2^{x_2}) \geq \log(p) - \lambda$

Hash Proof Systems

Key-encapsulation mechanisms with an additional property:

Knowing **sk**, can encapsulate in two modes

- **Valid:** Encapsulated key can be recovered
- **Invalid:** Encapsulated key is random

} computationally indistinguishable

Leakage reduces the min-entropy by at most λ, extract and mask the message

Our general construction:

Hash proof system + strong extractor

⬇

Key-encapsulation mechanism resilient to key leakage

Hash Proof Systems

Key-encapsulation mechanisms with an additional property:

Knowing **sk**, can encapsulate in two modes

- **Valid:** Encapsulated key can be recovered
- **Invalid:** Encapsulated key is random

} computationally indistinguishable

Leakage reduces the min-entropy by at most λ, extract and mask the message

Known instantiations:

- Decisional Diffie-Hellman
- Linear family (bilinear groups)
- Quadratic residuosity
- Composite residuosity (Paillier)

Extensions Satisfied By Our Schemes

Noisy leakage

- Leakage not necessarily of bounded length

$$H_\infty(sk \mid pk, leakage) > H_\infty(sk \mid pk) - \lambda$$

Leakage of intermediate values

- Once the keys are generated, are all intermediate values erased?
- Leakage depends on the random bits used for generating the keys
- Crucial for security under composition

Weak random source

- Keys generated using a low-entropy adversarially chosen source
- Need only a min-entropy guarantee for sk

Conclusions & Open Problems

- We can meaningfully model various forms of leakage
- We can build efficient schemes that resist them

- Leakage-resilient encryption from general assumptions?
  - From any CPA-secure scheme?
- Dealing with "iterative'' leakage and refreshed keys?
  - As in leakage-resilient stream-ciphers [DP08, P09]
- Other primitives? Other side channels?
  - Signature Scheme [KV09]
  - Bounded Retrieval Model [ADW09]
  - Hard-to-invert leakage [DKL09, KV09]