

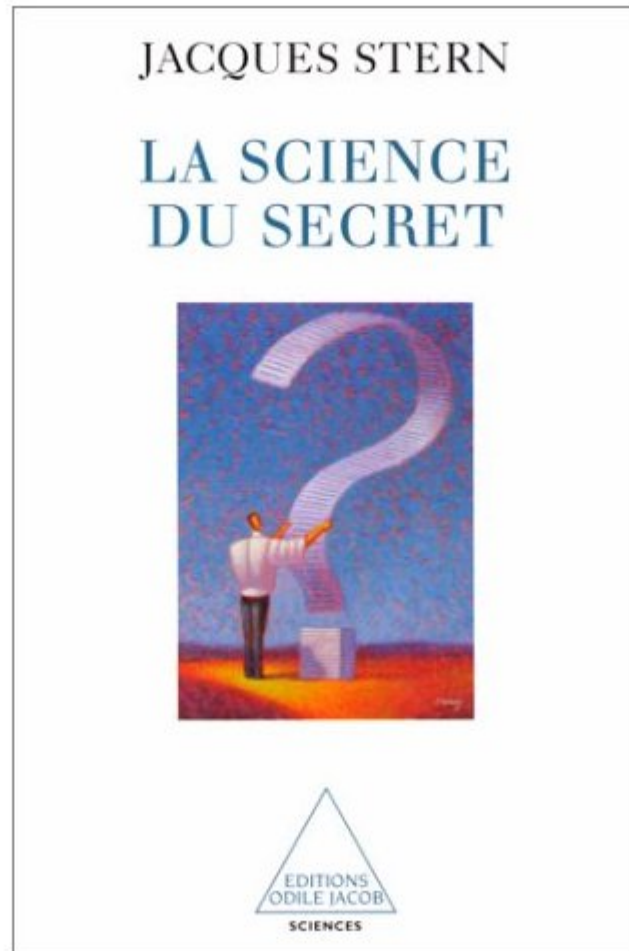
La Science du Secret sans Secrets

celebrating Jacques Stern's 60's birthday

Moti Yung

Columbia University and Google Research

Inspired by a Book by Jacques Popularizing Cryptography



- Doing research, teaching, applying in real world (business), popularizing, philosophizing deeply, reflecting on tradition in order to innovate, having the right mix of real engineering and mathematics in cryptography... is the path Jacques took in cryptography.
- We became scientific friends by noticing we share a lot scientific values and pragmatics... and “Cryptography” is part of our “Way of life”...

What am I going to Talk About?

- Modern cryptography started in the 1970's with cryptosystems (PKC, DES,...)
- I will talk about a personal view about one evolution path cryptosystems have taken.
- 70's to early 90's: Cryptosystems as part of a bigger system: **Crypto sub-systems considered the guards of the overall system** (i.e., performing encryption, authentication, identification)—
Crypto To the Rescue!!

Overview Continued..

- In the late 80's: Cryptosystems started to be embedded in “actual systems” and from mid 90's in “open systems” (Internet).
- Considered as “guards” and doing good for the system designer against malicious attacks.
- What was realized is that while guarding systems is ok task for cryptography, the cryptosystem itself needs to be guarded (computers are easy to break in). This is one big difference from military equipment and open systems.
- Breaking Machines easier than cryptanalyzing keys of proper sizes

So...

So

- What was realized is that while guarding systems is ok task for cryptography, **the cryptosystem itself needs to be guarded** (computers are easy to break in). This is one big difference from military equipment and open systems.
- Breaking Machines (i.e., hacking) easier than cryptanalyzing keys of proper sizes
- If the Cryptographic System are the system's "Guards," then
"Who will Guard the Guard Themselves?"
Quis custodiet ipsos custodes?
- If the secret is lost or partially lost, who will save crypto?

Key Exposure Problem

- The security of most cryptosystems relies crucially on some secret information (keys)
- What if these keys are lost, stolen, or otherwise exposed? (by hackers)
- In most application environments, key exposure (fully or partially) represents a very serious threat

Need to deal with...

- There is a need to protect the cryptosystem in their deployed environment, which is unsafe.
- Ways to do with this?
 - “Not my problem” approach: HW/SW engineers, please build better protected system !!!!
 - This is possibly too costly (systems are built to compute but not to secure themselves).
 - As I will point out, “seeming protection” by building special systems is not always the solution
 - Redesign the cryptography approach:
 - Secret sharing was a way suggested to protect keys in storage in the early days of cryptography

Cryptographers started to work on protection

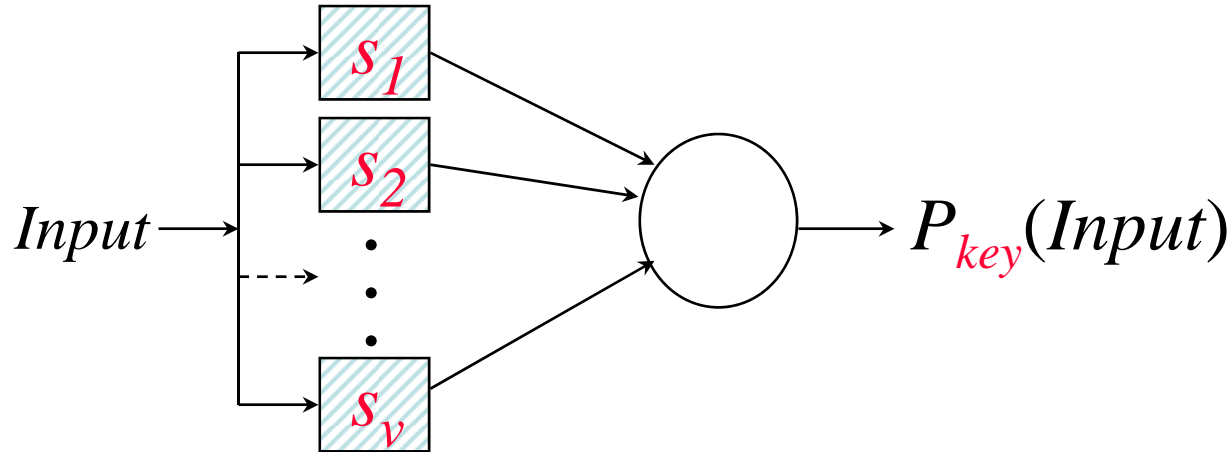
- We realized cryptographic **mathematics** is not enough: the way the system is storing the keys, the way attackers access the system, the physical properties of the systems— all need to influence the way we build systems
- So we take care of the Cryptographic back end by changing the crypto itself! (and **revise the mathematics** used!)
- I will review some approaches to deal with breaking of systems containing crypto keys.

(A) Threshold Cryptography

- Take a system and distribute it into devices: put the key pieces in many places
 - Availability (need only a threshold)
 - Protection: adversary learning less than the threshold pieces of the key, i.e. minority of shares only (security and robustness).
 - Proof of security: simulate the distributed system reducing its security to that of the non-distributed setting (RSA, ElGamal, etc.)

Distributed Cryptosystems

Function Sharing: [Boyd, CH,DF, F, DDFY94]



$t+1$ can compute $P_{key}(Input)$

t can not

no entity learns *key* after
function application

Robust: poly time
availability
for any misbehaving
minority t

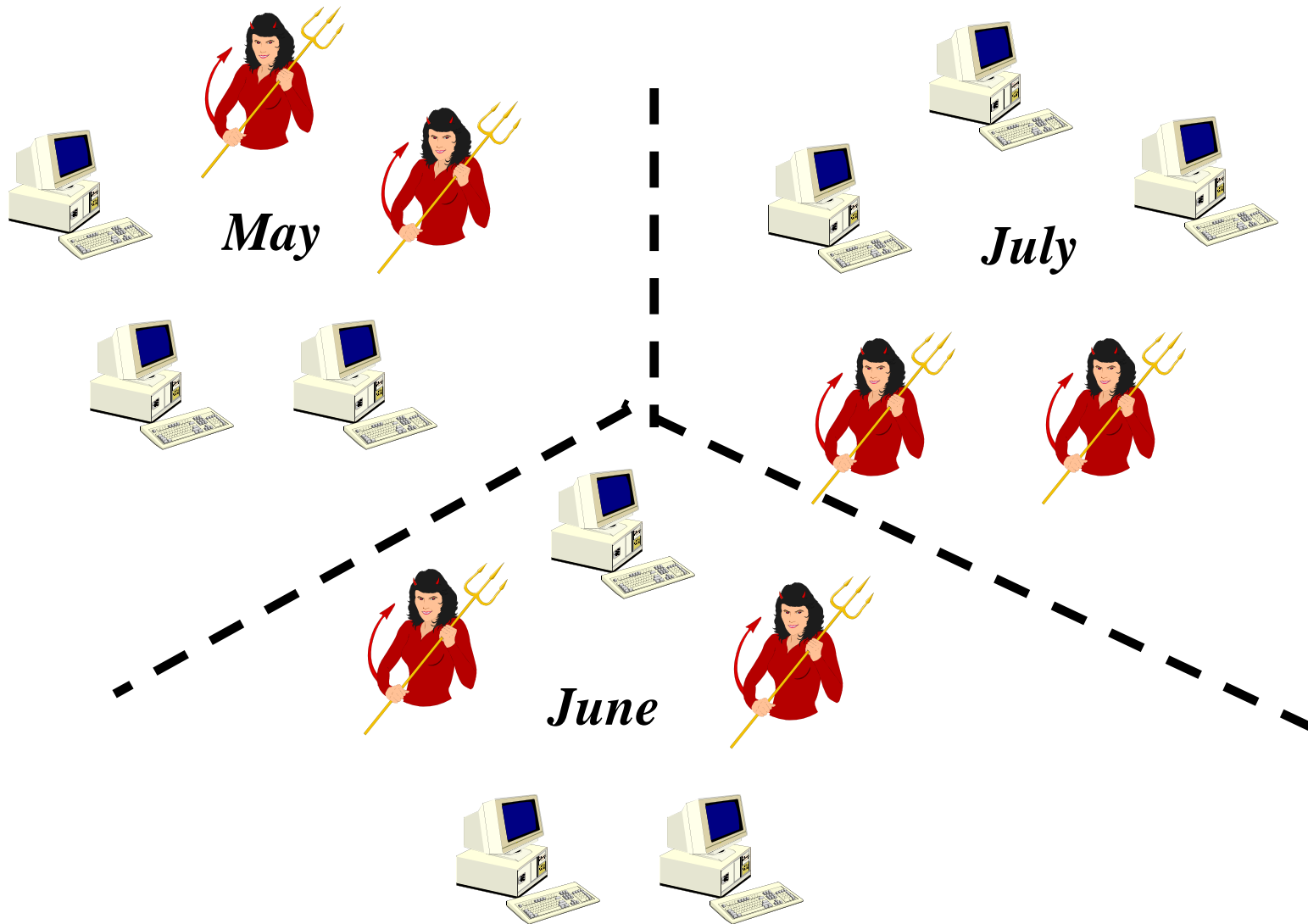
Simple RSA Example

- 3-out-of-3:
- Given a public key e , have the secret exponent d be split into $d_1+d_2+d_3=d$, but otherwise it is random.
- In RSA signing we get M and $M^d \pmod{N}$ and no more
- In distributed (space dimension protection) of the RSA signing: $M^{d_1} * M^{d_2} * M^{d_3} = M^d$
- Simulator: When signing M , $M^d \pmod{N}$ by the system, simulate the distributed system by producing two random d_1, d_2 (throughout) only, and then send out given the pair (M, M^d) : M^{d_1}, M^{d_2} and $M^d / (M^{d_1} * M^{d_2})$
- Note that this seems like simulator knows d but it does not.
- Jacques [FPS]: Threshold Paillier (that inspired one earlier work of mine, and I have used recently in a coming work), threshold RSA.....

(B) Proactive Security

- Ostrovsky & Y. 90: Add time dimension to the protection.
- Computers proactively every period re-randomize their keys, so attacker has only a period to get more than minority of shares (much harder task), previous period attacks that are not complete are not helpful.
- Secret sharing of pieces with re-sharing!! (redistribution at random)

Proactive security



T-Wise Re-Randomization

- Function (key) Sharing: a random polynomial with the secret at point 0:
 - Add a random polynomial Q with $Q(0)=0$.
 - More generally: take the polynomial and re-represent it with another random polynomial hiding the same secret.

(C) Protecting keys in a single location

- Distribution is not always possible
- Memory **Partial Leakage Resilient System:**
 - Hold the keys as shared in one location; this demonstrates that there are storage systems in which leaking part of the key does not leak the key (these are applications of secret sharing or Rivest 97 “all-or-nothing transform.”)
 - Usually implies adding redundancies (invokes coding theoretic techniques). One example where it is not (HIDE).

(D) Self-Protecting Systems

What to do with Full Leakage?? Looks devastating!

Mitigate it (compartmentalize damage)!

- Time is divided into N periods (e.g., months)
- Secret key stored on a device is dynamically updated over time
- Public key (when applicable) remains fixed
- Exposure of the key at period i affects only a limited number of other time periods
- End of period: update— for self protection

Like proactive but crypto operation in a single location

Different Paradigms

- (D1) Forward security
- (D2) Key-insulated security
- (D3) Intrusion-resilience

- Applicable to essentially *any* cryptographic functionality (public-/private- key authentication, encryption, signatures, etc.)

In Crypto 97: reflections on abstract (black box) mathematical crypto

Math Security is not enough

Physics and Trust Relations Matter:

- Paul Kocher: **side channel attack** (physical insecurity of secure system due to leakage). This started a huge area of important attacks on implementations.
- Young & Yung: **Kleptography**– physical protection hides the internals of the system, so manufacturer can attack the system (trust is needed in tamper proof system in the men behind the system– no scrutiny is possible). So over protection can protect the bad guys as well!

So: Trojan/Covert channels exist!!!
Malicious or by nature (physics)



Side Channel and Related Attacks

- Gave rise to a lot of engineering work.
- Exploit the power consumption, electromagnetic radiation, timing variations, etc., . . . of a cryptographic implementation (direct on device and through the network), resource sharing attacks: cash attacks, cloud computing attacks.
- For people in this area “provably secure black box cryptography” is essentially a myth!
- Smartcard and devices in general are big concern in France being an industrial leader on the issues (e.g., Ingenico).

Side Channel Attacks Are:

- **Powerful** (many attacks and specific countermeasures) but **device-specific**
 - successful attacks on real systems: Keyloq, RFID, etc.
- **Hard to evaluate**, hard to prevent in general
- Only a **part of the physical reality** (modeling physical reality into signals is hard and SCA and DRM suffer from this gap)
- Countermeasures, though very important, typically only increase the cost of attacks

Limitations of SCA practical works

- Good work but: Mainly rely on heuristics
- Use device-dependent metrics (e.g. variance)
- Use adversary dependent metrics (e.g. correlation)

Does this area that carefully takes care of engineering reality into account, have to live with the above limitations?

Alternatively: can we be more methodological about the issues?

(E) SCA issues that require “formal system model & theory”

- “How to compare two implementations?”
- “How to compare two adversaries?”

- Goal of the theoretical framework:
- Do Not give up on issues like the above!!

Thus: determine the extent to which these questions can be fairly answered

Suggested Formal System model and results

- In 04, Micali/Reyzin gave computational models with impossibilities (complexity theoretic model).
- Stantaert, Malkin, Yung: a “formal sc system oriented model” geared towards the engineering of systems: **it separates implementation** (modeled as communication channel) **and adversary** (modeled as channel access – receiver-- with some capabilities: probes and complexity limitations: various spec’s are possible). Advocated in the last three years.
- Possibilities and claims about measures and protection in a careful way.
- [Eurocrypt 09] We get formal results that can serve as base for engineering work and for understanding channels in general (systems-based model).

Nowadays

- The systems oriented model suggested motivated by engineering is a middleware in abstraction (systems based model).
- Recently (e.g., a full session in crypto 09): various “algorithmic models” that assume **leakage as part of the algorithmic steps** (less engineering oriented but interesting abstract analysis based on crisp definitions of the leakage as part of the algorithm: i.e., theoretical more abstract crypto theoretic approach).

Examples of Results

- Memory leakage but not the entire key → a CCA2 public key design
- Leakage at each step but not the entire step's state → pseudorandom generator (stream cipher)
- Etc.
- My view: all levels: fully abstract, system theory and engineering are useful methods to attack this problem.

Conclusions

- Dealing with protection of keys
 - In a single device or in many devices or in combination
 - Dealing with complete local leakage or partial global leakage or limited adversary on the global leakage
- Dealing in various settings: many devices, single device, single crypto-operation device
- The area is on-going and very excited work takes place nowadays
- Also: relations and bridging the gap with engineering is ongoing.

What have we learned?

- Dealing with engineering realities and physical realities (i.e., actual threats) does not have to be dealt with only locally or only by engineering
- Involve crypto even if problem is not cryptographic helps!
- The situation where secrets are lost is also part of the “science of secrets”

- Thank you Jacques for your friendship!
- Thank you all for listening!