iMinds
CONNECT.INNOVATE.CREATE

Software Security: Squaring the Circle?

Bart Preneel
COSIC – KU Leuven – Belgium
Firstname.Lastname(at)esat.kuleuven.be
http://homes.esat.kuleuven.be/~preneel
May 2015

1

Security goals

- Entity/device authentication
- Secure memory
- Secure update
- Data authentication (e.g. payment transactions)
- Secure boot
- Copy protection
- License enforcement
- Secure execution
- Protecting trade secrets

2

Outline

- Crypto
- Software versus hardware security
- The road ahead

3

Cryptography

- Cryptography has been very successful in terms of scientific methodology
  - elegantly solving a narrow problem
- Moving problems to keys and mathematical assumptions

4

Disclaimer: security ≠ cryptography

Most systems break elsewhere

Cryptography is more bypassed than broken

But if crypto is broken, there is trouble

5

Crypto history

- pre-1915: manual encryption or simple devices

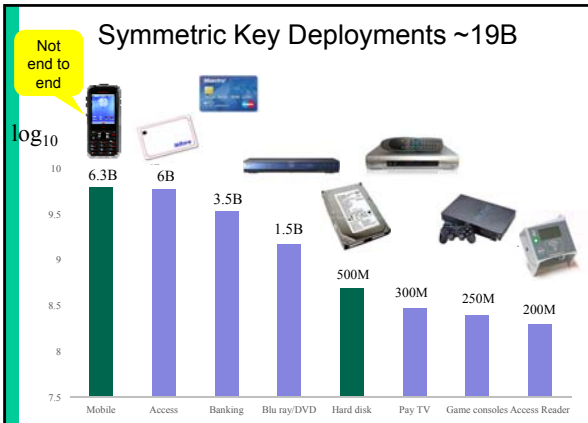
- 1915: rotor machines: (electro-)mechanical

- 1960's: electronic encryption
- 1975: integrated hardware
- 1990: software
- 2015: everywhere

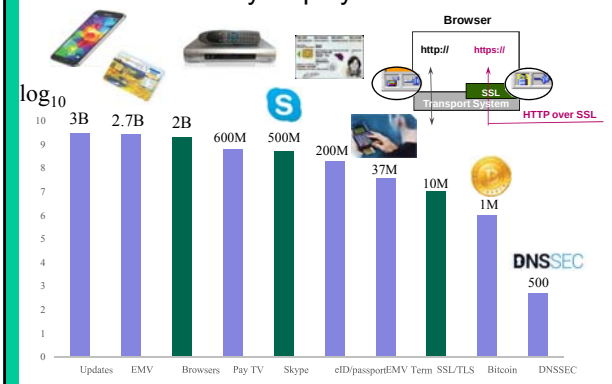
Deployment of cryptography

- most crypto in volume and market serves for data and entity authentication
  - code updates
  - payments: credit/debit/ATM/POS and SSL/TLS
- confidentiality
  - government/military secrets
  - DRM/content protection
  - telco: not end-to-end or with a backdoor
  - hard disk encryption: backdoored?
  - ehealth (growing market)
  - most data in the cloud is not encrypted

Symmetric Key Deployments ~19B



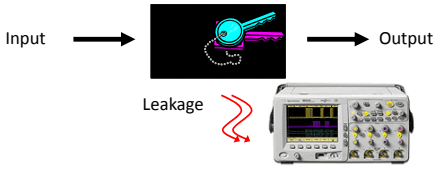
Public Key Deployments ~9B



No figures for VPN market – a few million?

Side-Channel Leakage

- Physical attacks ≠ Cryptanalysis
  (gray box, physics) (black box, maths)
- Does not tackle the algorithm's mathematical security

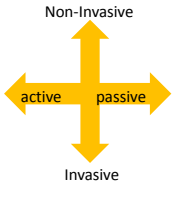


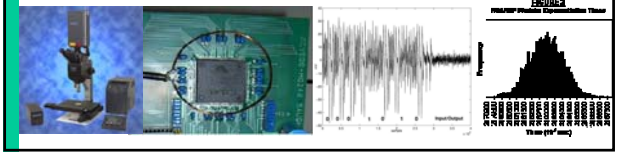
- Observe physical quantities
- Modify the data or algorithm during execution

10

Classification of Physical Attacks

- Active versus passive
  - active: perturbate and conclude
  - passive: observe and infer
- Invasive versus non-invasive
  - invasive: open package and contact chip
  - semi-invasive: open package, no contact
  - non-invasive: no modification





Cryptography

- Needs next to mathematical also physical assumptions
  - authenticated copies of public keys
  - keep secret and private keys secret
  - guarantee correct execution of algorithm
  - limited leakage of internal variables

12

Cryptography in software

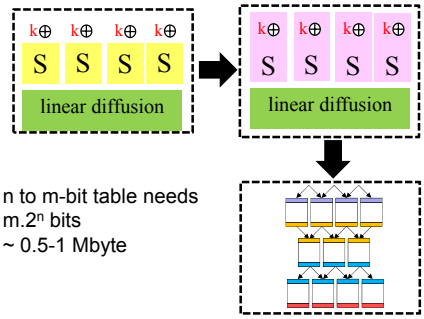
In software implementations the attacker typically has full access to the device: whitebox attack model

Easy to find the key [Shamir-van Someren'98]

13

Whitebox cryptography

Implement block cipher in such a way that it is hard to extract the key [Chow-Eisen-Johnson-van Oorschot'02]

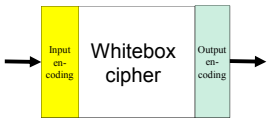


n to m-bit table needs
$m.2^n$ bits
~ 0.5-1 Mbyte

14

Whitebox cryptography (2)

- Attacker can always evaluate the function
  - need input and output encodings: problem is shifted to secrets in other places



- Limitations: copy the code to another device
- Other idea: insert code to be executed into the look-up tables

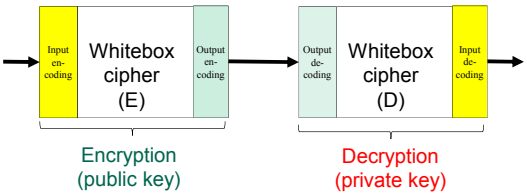
15

Practical whitebox constructions

- Several attempts in the literature (DES, AES) between 2002 and 2010
- By early 2014 they were all broken

Several companies still have proprietary designs

Does whitebox cryptography yield an efficient public-key encryption scheme?



Answer: no ☹

- table lookups can be reversed layer by layer

But strong obfuscation would yield such a scheme [Hellman, ca 75]

17

Code Obfuscation*: Goal

Make programs "unintelligible" while maintaining their functionality

- example from Wikipedia:

```
@P=split//,".URRUU\c8R";@d=split//,"\nrekcah xinU / lreP
rehtona tsuJ";sub p{
@p{"r$p","u$p"}=(P,P);pipe"r$p","u$p";++$p;($q*=2)+=$f=!fork
;map{$P=$P[$f^ord ($p{$_})&6];$p{$_}=/
^$P/ix?$P:close$_}keys%p}p;p;p;p;p;map{$p{$_}=-/^[P.]/&&
close$_}%p;wait until$?;map{/^r/&&<$_>}%p;$_=$d[$q];sleep
rand(2)if/\S/;print
```

* slides on obfuscation inspired by/borrowed from: Shai Halevi
http://people.csail.mit.edu/shaih/pubs/IndistinguishabilityObfuscation.pptx

18

Code Obfuscation: Why

- Hiding cryptographic keys
- Hiding license checks
- Prevent cheating in online games
- Hiding nature of 0-days that are being patched
- Hiding algorithm
  - example: assume you have a winning strategy for chess

19

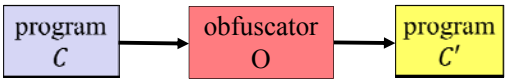
Code Obfuscation

- Widely used in practice: art or science?
  - CFG flattening, virtual machines, data obfuscation,…
  - some rules-of-thumb, sporadic tool support
  - *Wikipedia: "At best, obfuscation merely makes it time-consuming, but not impossible, to reverse engineer a program"*
- Can this be made scientific?

20

Defining Obfuscation

- An efficient public procedure O(•)
  - everything is known about it
  - except the random coins that it uses
- Takes as input a program C
  - e.g., encoded as a circuit
- Produces as output another program C'
  - C' computes the same function as C
  - C' at most polynomially larger than C
  - C' is "unintelligible": defining this is tricky



21

What's "Unintelligible"? Attempt 1

- *Ideally: cannot do much more with C' than running it on various inputs*
- **Virtual Black Box Obfuscation (VBB):** anything that can be efficiently computed from C' can be efficiently computed given oracle access to C

22

What's "Unintelligible"? Attempt 1

- **VBB:** If C depends on some secrets that are not readily apparent in its I/O, then C' does not reveal these secrets
- generic VBB is impossible [Barak+01]
  - **Thm:** If PRFs exist, then there exists PRF families $F = \{f_s\}$, for which it is possible to recover s from *any circuit* that computes f_s.
    - these PRFs are *unobfuscatable*

23

Unintelligible: Virtual Black Box

For a few functions, VBB obfuscation is feasible
E.g. point-functions/cryptographic locks

$$f_{a,b}(x) = \begin{cases} b & \text{if } x = a \\ \perp & \text{otherwise} \end{cases}$$

Even if there are extensions, it is clear that we need to weaken the requirements if we want a universal obfuscator (i.e. works for every function)

24

What's "Unintelligible"? Attempt 2 Indistinguishability Obfuscation (iO)

- If C_1, C_2 compute the same function (and $|C_1| = |C_2|$) then $O(C_1) \approx O(C_2)$ [Barak+01]:
  - Indistinguishable even if you know C_1, C_2
- Note: inefficient *iO* is always possible
  - $O(C)$ = lexicographically 1st circuit computing the same function as C

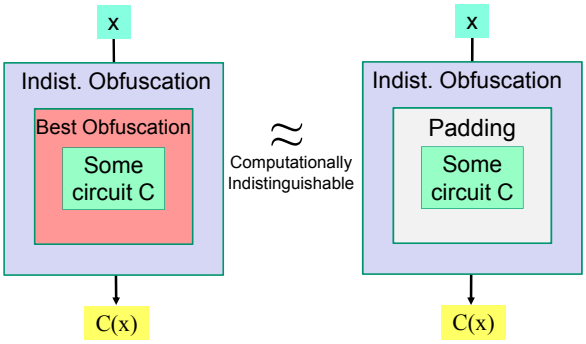
(canonical form)

  - Canonicalization is inefficient (unless P=NP)

25

Best-Possible Obfuscation [Goldwasser-Rothblum'07]

Efficient iO ⇒ hides as much about the input circuit as any circuit of a given size



Many Applications of iO

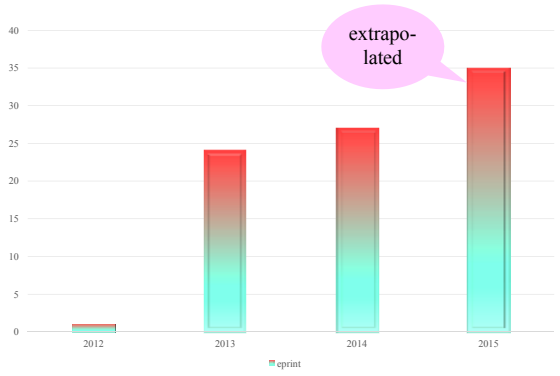
- AES → public key encryption [GGH+13, Sahai-Waters'14]
- Witness encryption: encrypt x so only someone with proof of Riemann Hypothesis can decrypt [Garg-Gentry-Sahai-Waters'13]
- Deniable encryption [Sahai-Waters'14]
- Functional encryption: noninteractive access control [GGH+13], $Dec(Key_y, Enc(x)) \rightarrow F(x, y)$
- …

"central hub in cryptography"

Constructions of iO

- [GGH+13]= [Garg-Gentry-Halevi-Sahai-Raikova-Waters] FOCS 2013
  - iO for NC^1 (polynomial-size, log-depth circuits) using multi-linear jigsaw problem
  - iO for general functions: NC^1 construction + Fully Homomorphic Encryption
- Follow-up work to streamline assumptions, e.g. [Pass-Seth-Telang'14]

Number of obfuscation papers on IACR eprint



29

Practical? Constructions of iO

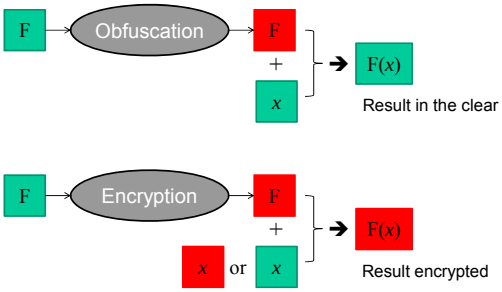
- [Banescu-Ochoa-Kunze-Pretschner'15] implement [GGHSRW'13] scheme
- 2-bit multiplier
  - 4 inputs bits
  - 1-8 AND gates per input bit
- Generating one obfuscation takes 10^{27} years on a 2.6 GHz PC
- Implementation in memory: 20 Zetabyte (10^{21})
- Circuit evaluation: 1.8 $.10^8$ years

Limitations of iO

- No guarantees on leakages of information
- No black box construction of a **collision resistant hash function** with a polynomial security loss from a general purpose iO and a one-way (or trapdoor) permutation [Asharov-Segev'15]

- Note that obfuscation only intends to resist **passive** attackers

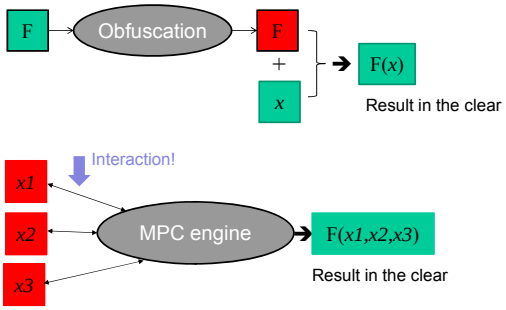
Obfuscation ⇔ Homomorphic Encryption



FHE = infeasible but somewhat FHE feasible today

32

Obfuscation ⇔ Multiparty Computation



MPC feasible for more and more functions

33

Software security

Firenze
Piazza Santa
Maria Novella

34

Software security

"it's turtles all the way down"

preacher Joseph Frederick Berg (1854):

- My opponent's reasoning reminds me of the heathen, who, being asked on what the world stood, replied, "On a tortoise." But on what does the tortoise stand? "On another tortoise." With Mr. Barker, too, there are tortoises all the way down.

35

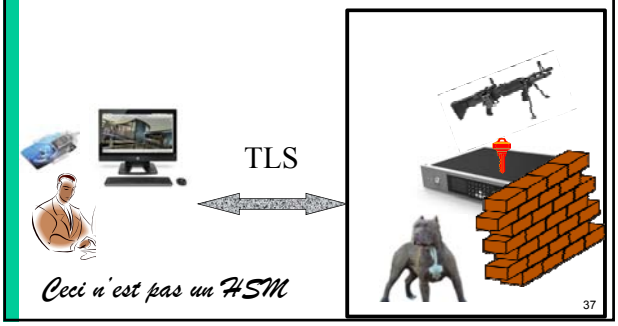
Need for hardware security

- Integrity protected storage for hash value or public key
- Secure key storage
- AES instruction
- RNG
- TPM or smart card (SIM)
- HSM

Trustworthy subsystems that can be certified

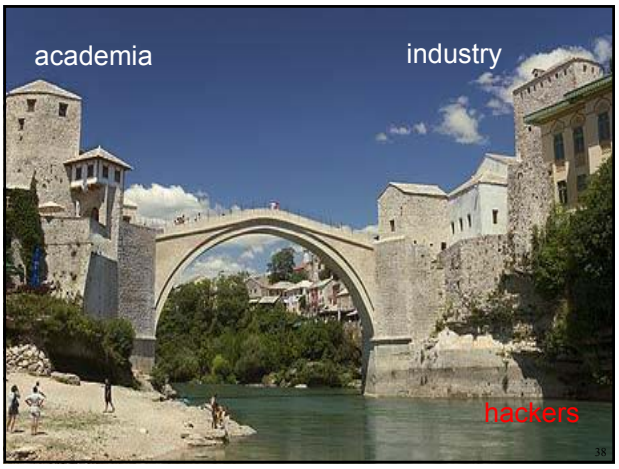
Secure hardware pitfall

- it is great to have secure hardware
- but how do you authenticate who/what can talk to it?



Ceci n'est pas un HSM

37



38

Symmetric cryptography: block ciphers, hash functions….

Design: easy	Attack models: clear
Security evaluation: complex clear attack models and metrics most attacks impractical	Performance evaluation: reproducible efficient tools
Tools for security evaluation: few are shared	Open competition AES-NESSIE-eSTREAM-SHA3-CAESAR

39

Side channel attacks: DPA, CPA, Higher Order DPA, Mutual Information Analysis

Design: hard	Attack models: disputed
Security evaluation: very complex time consuming platform dependent	Performance evaluation: platform dependent some tools
Tools for security evaluation: some developments DPA tools, SASEBO CC JIL = confidential	DPA competition: moderate success Industry switched to security by obscurity Academia: toy implementations

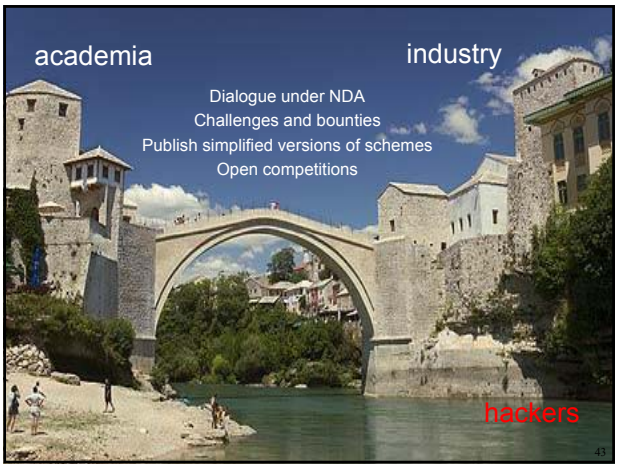
40

Software security: obfuscation

Design: very hard	Attack models: +/- clear
Security evaluation: manual and ad hoc most schemes easy to break	Performance evaluation: platform dependent some tools
Tools for security evaluation: few	Industry still in security by obscurity model

41

42



Challenges (1): industry

- security by obscurity: is this scientific? (August Kerckhoffs)
  - risk of backdoors
- commitment to help bridging the gap
- need to think about certification
  - in collaboration with academia
  - which information about the certification is public?
  - how is information shared over the chain from vendor to customer to end user?
  - what about backdoors?

Challenges (2): academia

1. Metrics
2. Metrics
3. Metrics

Scientific approach
- Measurable
- Reproducible
- Tools

Cannot burn many person-months of graduate students on reverse engineering

Challenges (3): hackers

Strive for security based on mathematics that is hard to bypass

[Bill Neugent]
Most cryptosystems are secure because most people would rather eat liver than do mathematics

Challenges (4): collaboration

- industry: explore collaboration and sharing methods
- academia: evaluate realistic implementations with combined countermeasures
- develop joint solutions, tools and evaluation methods

The end

Thank you for your attention