La Cryptographie

Pascal Junod // HEIG-VD

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Plan

- Cryptographie symétrique
- Cryptographie asymétrique
- Chiffrement par bloc
- Chiffrement par flot
- Signatures digitales
- Courbes (hyper-)elliptiques
- Cryptanalyse statistique
- Couplages de Weil
- Factorisation de grands nombres
- Cryptographie quantique
- Cryptographie post-quantique

Plan

$$\mathrm{SK}^{(i)} = \frac{e(\mathrm{hdr}_{i,0}, g_1^{r(\beta+s_u)})}{\prod_{j=1}^{M}\left(\mathrm{SK}_j^{(i)}\right)^{s_u}}$$

$$\mathrm{hdr}_{i,j} = \left(g^{rt_j^{(i)}}, \left(v^r \cdot g_{n+1-\phi}^r\right)^{t_j^{(i)}}\right) \in \mathbb{G}^2$$

$$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$$

$$\mathbb{G} = \{\xi(x) : x \in \mathbb{Z}_p^+\}$$

$$\mathbf{Encrypt}(\mathsf{ek}, \mathbb{A})$$

$$\mathrm{SK} = \frac{e(\mathrm{hdr}_0, g_1^{r(\beta+s_u)})}{\prod_{i=1}^{N} \mathrm{SK}_i^{s_u}}$$

$$\mathbb{A} = \beta_1 \wedge \beta_2 \wedge \ldots \wedge \beta_i \wedge \ldots \wedge \beta_N$$

$$P, Q \in \mathbb{F}_p[X_1, \ldots, X_n]^s$$

$$\mathrm{Adv}^{\mathrm{ind}}(\lambda, n, \mathfrak{B}(u_i)_{1 \le i \le \ell}, \mathcal{A}) = |2\Pr[b = b'] - 1|,$$

$$\mathrm{hdr}^{(i)} = \left(g_n^{t^{(i)}}, \mathrm{hdr}_{i,1}, \ldots, \mathrm{hdr}_{i,M}\right)$$

$$(\varphi(p_1(X_1, \ldots, X_n)), \ldots, \varphi(p_s(X_1, \ldots, X_n)))$$

$$\left(g^{P(X_1,\ldots,X_n)}, \left(h^{Q(X_1,\ldots,X_n)} \prod_{j \in \beta_i} g_{n+1-j}^{r}\right)\right) \in \mathbb{G}^s \times \mathbb{G}_T^s$$

$$\mathrm{hdr}_i = \left(g^{rt_i}, \left(v^r \prod_{j \in \beta_i, j \neq k} g_{n+1-j+k}^{t_i}, C_0\right)\right) \in \mathbb{G}^2$$

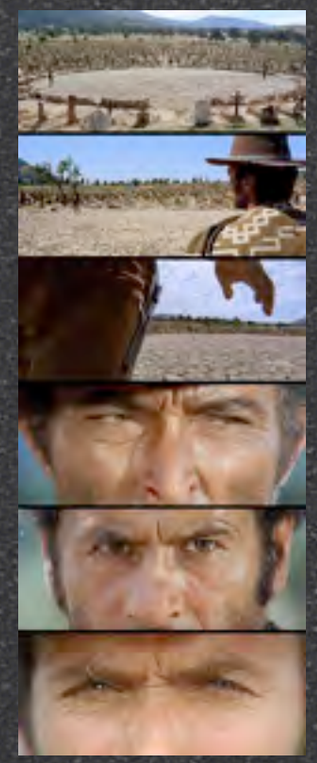
$$\mathrm{Adv}^{\mathrm{GDHE}}(\mathcal{A}) < \frac{(q+2(4n+6+2R)+2)^2}{p}$$

La Cryptographie

et matériel embarqué: Il buono, il brutto, il cattivo

Pascal Junod // HEIG-VD

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud



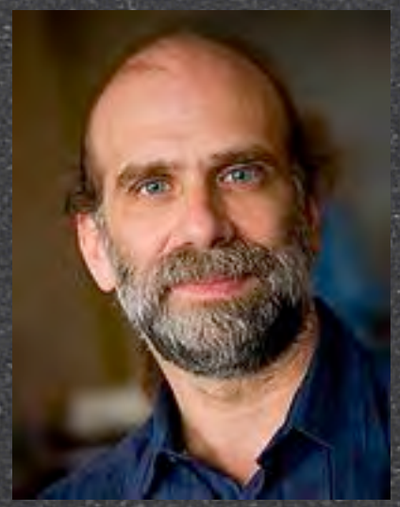
Plan

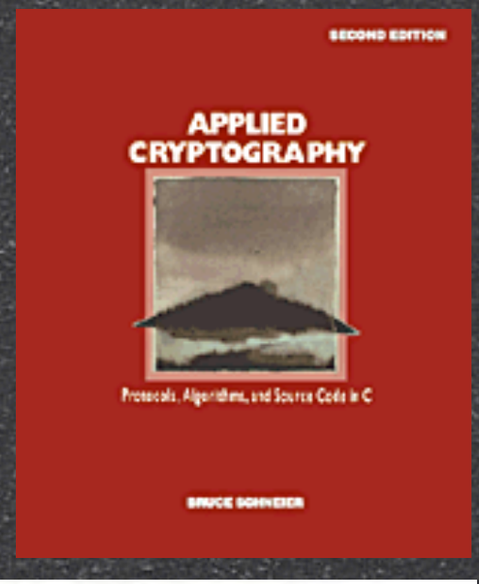
- Cryptographie et Sécurité
- Attaques par «Side-Channel»
- OpenSSL et Consoeurs

Cryptographie et Sécurité

De la Crypto Partout

- Avec l'apparition de l'Internet et d'autres réseaux, l'utilisation de la cryptographie a explosé.





This new edition of the cryptography classic provides you with a comprehensive survey of modern cryptography. The book details how programmers and electronic communications professionals can use cryptography -- the technique of enciphering and deciphering messages -- to maintain the privacy of computer data. It describes dozens of cryptography algorithms, gives practical advice on how to implement them in cryptographic software, and shows how they can be used to solve security problems. Covering the latest developments in practical cryptographic techniques, this new edition shows programmers who design computer applications, networks, and storage systems how they can build security into their software and systems.

Mais...

Software Error Category: Porous Defenses

[5] CWE-285: Improper Access Control (Authorization)

If you don't ensure that your software's users are only doing what they're allowed to, then attackers will try to exploit your improper authorization and...MORE >>

[6] CWE-807: Reliance on Untrusted Inputs in a Security Decision

Driver's licenses may require close scrutiny to identify fake licenses, or to determine if a person is using someone else's license. Software developers...MORE >>

[10] CWE-311: Missing Encryption of Sensitive Data

If your software sends [redacted] private data or authentication credentials, that information...MORE >>

[11] CWE-798: Use of Hard-coded Credentials

Most of the CWE Top [redacted] mistake; for this issue, though, customers...MORE >>

[19] CWE-306: Missing Authentication for Critical Function

In countless action [redacted] crawling through heating ducts...MORE >>

[22] CWE-732: Incorrect Permission Assignment for Critical Resource

If you have critical programs, data stores, or configuration files with permissions that make your resources accessible to the world - well, that's just what they'll become...MORE >>

[24] CWE-327: Use of a Broken or Risky Cryptographic Algorithm

You may be tempted [redacted] difficult for attackers to crack. This kind of grow-your-own cryptography is a welcome sight to attackers...MORE >>

Source: http://www.sans.org/top25-software-errors

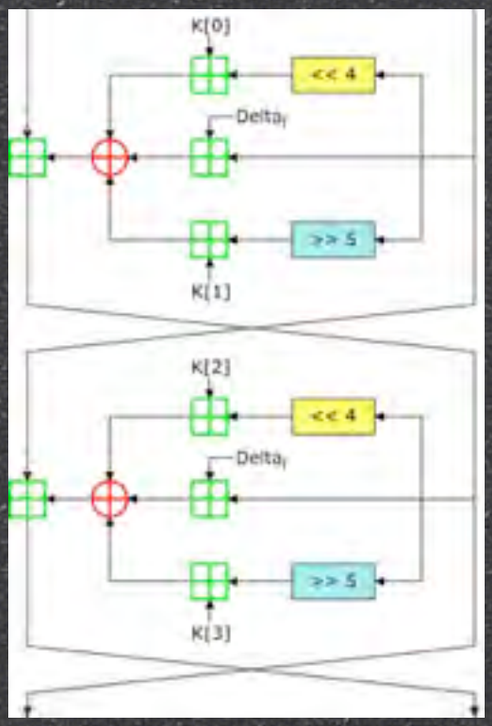
Pourquoi se Plante-t-on ?

- Mauvais choix de primitives
  - TEA et le hack de la XBOX
  - RC4 et WEP
  - MD5

TEA et le Hack de la XBOX

TEA et le Hack de la XBOX

- TEA utilisé comme fonction de compression dans un algorithme de hachage «maison» utilisée pour authentifier le code durant le boot.



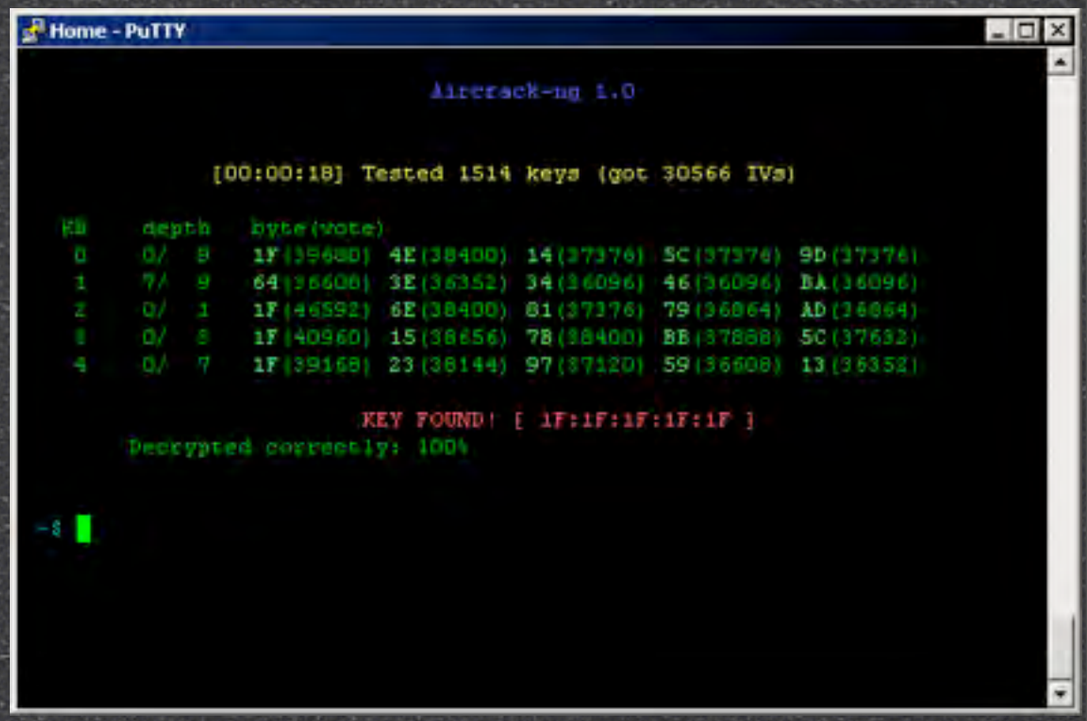
- Pas de chance: en mode «hachage», clefs équivalentes == collisions

Best public cryptanalysis

TEA suffers from equivalent keys (Kelsey et al., 1996) and can be broken using a related-key attack requiring 2^{23} chosen plaintexts and a time complexity of 2^{32}.[1]

RC4 et WEP

- RC4 utilisé comme chiffrement à flot dans le standard sans-fil WEP.
- Pas de chance: RC4 souffre de nombreuses imperfections statistiques au début de sa sortie...



MD5

- MD5 reste un des algos de hash les plus utilisés en pratique
- Pas de chance: il a été complètement cassé en 2004 (résistance aux collisions)

Chosen-prefix Collisions for MD5 and Colliding X.509 Certificates for Different Identities

Marc Stevens[1], Arjen Lenstra[2], and Benne de Weger[1]

[1] TU Eindhoven, Faculty of Mathematics and Computer Science
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
[2] EPFL IC LACAL, Station 14, and Bell Laboratories
CH-1015 Lausanne, Switzerland

Abstract. We present a novel, automated way to find differential paths for MD5. As an application we have shown how, at an approximate expected cost of 2^{50} calls to the MD5 compression function, for any two chosen message prefixes P and P', suffixes S and S' can be constructed such that the concatenated values $P\|S$ and $P'\|S'$ collide under MD5. Although the practical attack potential of this construction of *chosen-prefix collisions* is limited, it is of greater concern than random collisions for MD5. To illustrate the practicality of our method, we constructed two MD5 based X.509 certificates with identical signatures but different public keys *and* different Distinguished Name fields, whereas our previous construction of colliding X.509 certificates required identical name fields. We speculate on other possibilities for abusing chosen-prefix collisions. More details than can be included here can be found on **www.win.tue.nl/hashclash/ChosenPrefixCollisions/**.

Pourquoi se Plante-t-on ?

- Mauvais choix de protocole
  - IPsec en mode «encrypt-only»

IPsec en Mode «Encrypt-Only»

- On peut configurer le protocole IPsec de (trop) nombreuses manières:
  - «Encrypt-Only»
  - «Authenticate-Only»
  - «Encrypt-and-Authenticate»

IPsec en Mode «Encrypt-Only»

Attacking the IPsec Standards in Encryption-only Configurations

Jean Paul Degabriele[1] and Kenneth G. Paterson[2] *

[1] Hewlett-Packard Laboratories, Bristol
Filton Road, Stoke Gifford, Bristol BS34 8QZ, UK.
jeanpaul.degabriele@gmail.com
[2] Information Security Group,
Royal Holloway University of London,
Egham, Surrey TW20 0EX, UK.
Kenny.Paterson@rhul.ac.uk

Abstract. At Eurocrypt 2006, Paterson and Yau demonstrated how flaws in the Linux implementation of IPsec could be exploited to break encryption-only configurations of ESP, the IPsec encryption protocol. Their work highlighted the dangers of not using authenticated encryption in fielded systems, but did not constitute an attack on the actual IPsec standards themselves; in fact, the attacks of Paterson and Yau should be prevented by any standards-compliant IPsec implementation. In contrast, this paper describes new attacks which break *any* RFC-compliant implementation of IPsec making use of encryption-only ESP. The new attacks are both efficient and realistic: they are ciphertext-only and need only the capability to eavesdrop on ESP-encrypted traffic and to inject traffic into the network. The paper also reports our experiences in applying the attacks to a variety of implementations of IPsec, and reflects on what these experiences tell us about how security standards should be written so as to simplify the task of software developers.

Keywords: IPsec, integrity, encryption, ESP, standard.

Pourquoi se Plante-t-on ?

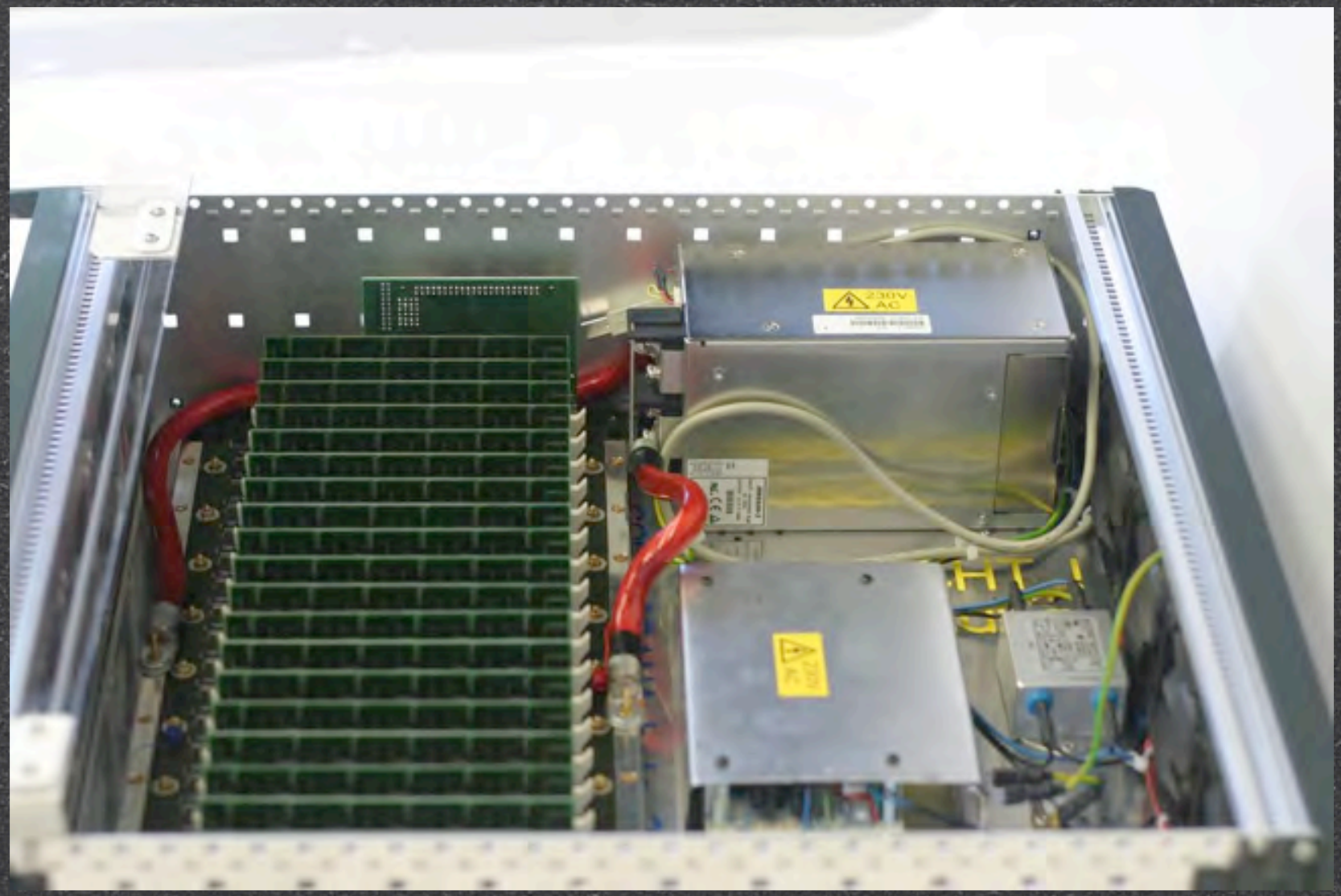
- Mauvais choix de taille de clefs:
- Longueurs de clefs «export-compatible»
- Authentification du code des calculatrices TI-x

Longueurs de clefs «export-Compatible»

- TLS 1.0 cipher suites «super-cool»:

TLS_RSA_EXPORT_WITH_RC4_40_MD5	*	RSA_EXPORT	RC4_40	MD5
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5	*	RSA_EXPORT	RC2_CBC_40	MD5
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA	*	RSA_EXPORT	DES40_CBC	SHA
TLS_RSA_WITH_DES_CBC_SHA		RSA	DES_CBC	SHA
TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA	*	DH_DSS_EXPORT	DES40_CBC	SHA
TLS_DH_DSS_WITH_DES_CBC_SHA		DH_DSS	DES_CBC	SHA
TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA	*	DH_RSA_EXPORT	DES40_CBC	SHA
TLS_DH_RSA_WITH_DES_CBC_SHA		DH_RSA	DES_CBC	SHA
TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA	*	DHE_DSS_EXPORT	DES40_CBC	SHA
TLS_DHE_DSS_WITH_DES_CBC_SHA		DHE_DSS	DES_CBC	SHA
TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA	*	DHE_RSA_EXPORT	DES40_CBC	SHA
TLS_DHE_RSA_WITH_DES_CBC_SHA		DHE_RSA	DES_CBC	SHA
TLS_DH_anon_EXPORT_WITH_RC4_40_MD5	*	DH_anon_EXPORT	RC4_40	MD5
TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA		DH_anon	DES40_CBC	SHA
TLS_DH_anon_WITH_DES_CBC_SHA		DH_anon	DES_CBC	SHA

Longueurs de clefs «export-Compatible»



Source: http://www.copacobana.org

TI-x Secure Boot & RSA

TI-83 Plus OS Signing Key Cracked
Posted by Michael on 31 July 2009, 15:33 GMT

The ever-mysterious Benjamin Moody posted a cryptic message on the United-TI forum yesterday. In it, he listed the factorization of the 512-bit RSA modulus used by TI's OS signing key for the 83+ (the "0004 key"). No other details are yet available about how he achieved this feat of substantial brute forcing power. In the event of United-TI downtime, Brandon Wilson has put a copy of Benjamin's values on his personal website.

With this achievement, any operating system can be cryptographically signed in a manner identical to that of the original TI-OS. Third party operating systems can thus be loaded on any 83+ calculators without the use of any extra software (that was mentioned in recent news) Complete programming freedom has finally been achieved on the TI-83 Plus!

Update: Benjamin has posted additional details on the United-TI forum thread.

Update: A distributed computing project has been set up. Information about how to join the effort to crack the OS keys for the remaining TI models can be found here.

Reply to this article

TI-x Secure Boot & RSA


Fun Number Theory Facts

FloppusMaximus #1

Advanced Member

Group: Members
Posts: 462
Joined: 23-August 08
Gender: Male

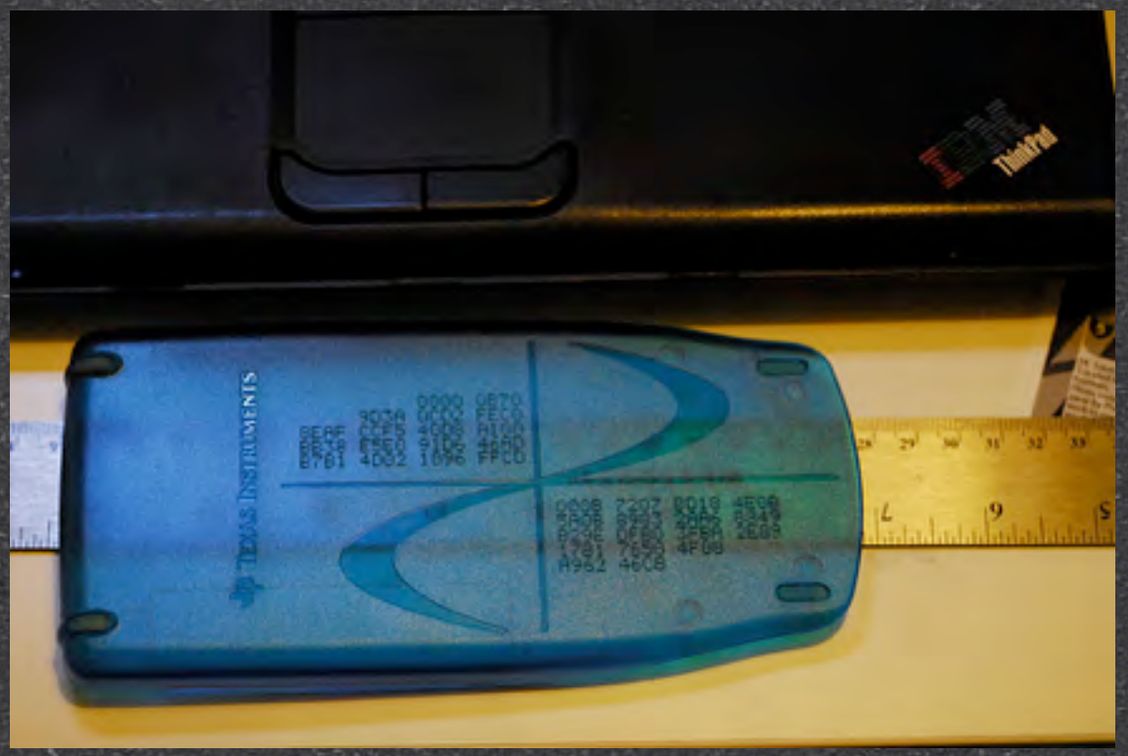
Posted 30 July 2009 - 04:07 AM

Dear community,

I have been politely asked to remove the former contents of this post.

This post has been edited by **FloppusMaximus**: 27 August 2009 - 12:51 AM

Reply MultiQuote



TI-x Secure Boot & RSA

FloppusMaximus #18

Advanced Member

Group: Members
Posts: 462
Joined: 23-August 08
Gender: Male

Posted 31 July 2009 - 07:32 PM

Whoa! OK, let's take them one at a time.

How did I do this? With the best tools I could find for the job. The best algorithm for factoring really large general numbers (i.e., numbers without any special properties) is the general number field sieve. The best currently-available implementation of the GNFS consists of a combination of the GGNFS and Msieve projects. It's really the guys behind these tools who deserve the credit for making this possible. While it does take a bit of work to get the tools set up correctly, most of what I did was sitting around waiting for it to finish, and every once in a while, telling the script to try another filtering run.

Some fun statistics:
- The factorization took, in total, about 1745 hours, or a bit less than 73 days, of computation. (I've actually been working on this since early March; I had a couple of false starts and haven't been able to run the software continously.)
- My CPU, for reference, is a dual-core Athlon64 at 1900 MHz.
- The sieving database was 4.9 gigabytes and contained just over 51 million relations.
- During the "filtering" phase, Msieve was using about 2.5 gigabytes of RAM.
- The final processing involved finding the null space of a 5.4 million x 5.4 million matrix.

Oh, and how long have I had this? About two days now. The job finished on Wednesday afternoon, I tested out the result with PongOS, then I came here to tell you all about it.

The other keys will come in their time, I'm sure. If anybody else would like to try factoring one of them, just let me know so we don't step on each other's toes. I haven't started working on any of them yet; I think I'll probably try 0102 next, but I could be persuaded otherwise.

I'm still rather amused that this happened at almost the same time Free83P was released. Great minds think alike...

This post has been edited by **FloppusMaximus**: 31 July 2009 - 07:33 PM

Pourquoi se Plante-t-on ?

- Mauvaise utilisation de la crypto:
  - M$ Lan Manager Hash
  - Utilisation incorrecte de RSA
    - Chiffrement d’une clef symétrique sans «padding» avec un petit exposant 8-))
    - Signature sans «pre-processing» du message
    - ...

What might go wrong ?

- Et la liste des horreurs est loin d’être terminée !!!

Heureusement...

- On connaît des primitives correctement conçues et analysées:
  - Chiffrement: AES (NIST FIPS 192)
  - Hachage: SHA-2 (NIST FIPS 180-2)
  - Chiffrement à clef publique et signatures: RSA-OAEP et RSA-PSS (PKCS #1, v2.1)
  - Mise au point de clef: ECDH (NIST SP800-56A)
  - Protocoles: TLS (RFC 5246)

Le Joe Codeur moyen a-t-il une chance de s’en tirer ?

Pascal Junod -- La Cryptographie

Insomni’hack 2011, 4 mars 2011, Genève, Suisse

Heuuuuuu...

Attaques de type «Side-Channel»

Adversaires «Black-Box»

- C’est la définition privilégiée des cryptologues

ceux qui n`utilisent leur ordinateur que pour faire du mail et écrire des papiers en LaTeX ;-)

Adversaires «Black-Box»

- Je modélise mon algorithme/protocole/système comme un ensemble d'oracles
- Interaction avec ces oracles:
  - Chiffré
  - Texte clair connu
  - Texte clair/chiffré choisi (adaptatif ou non)

Adversaires «Black-Box»

- Je prouve (mathématiquement) que mon algorithme/protocole/système est sûr si les primitives cryptographiques utilisées sont sûres.
- Exemples:
  - RSA-OAEP
  - RSA-PSS

Adversaires «Grey-Box»

- Adversaires pas prévus par les cryptologues (théoriciens)
- Ils interagissent avec les primitives cryptographiques, mais ils obtiennent en plus un tout petit peu d'information sur le calcul:
  - Timings
  - Fuites physiques
  - Fautes

Adversaires «White-Box»

- Le cauchemar de nombreux cryptologues

- Peuvent faire TOUT ce qu'ils souhaitent
- «Reverse-engineering» du SW/HW
- Lire/écrire toutes les mémoires, y-compris celles contenant des secrets
- Perturber tous les calculs

Plates-Formes Embarquées

- Petits objects:
  - Modules USB
  - Smartcards
  - Chips
  - PCs embarqués
  - ...

Plates-Formes Embarquées

- Principales caractéristiques des plates-formes embarquées pour un adversaire:
  - Locales
  - Lentes
  - Peu chères
  - ...

Attaques par «Side-Channel»

- Timing
- Fuites physiques
- Fautes

Attaques par «Timing»

Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems

Paul C. Kocher

Cryptography Research, Inc.
607 Market Street, 5th Floor, San Francisco, CA 94105, USA.
E-mail: paul@cryptography.com.

Abstract. By carefully measuring the amount of time required to perform private key operations, attackers may be able to find fixed Diffie-Hellman exponents, factor RSA keys, and break other cryptosystems. Against a vulnerable system, the attack is computationally inexpensive and often requires only known ciphertext. Actual systems are potentially at risk, including cryptographic tokens, network-based cryptosystems, and other applications where attackers can make reasonably accurate timing measurements. Techniques for preventing the attack for RSA and Diffie-Hellman are presented. Some cryptosystems will need to be revised to protect against the attack, and new protocols and algorithms may need to incorporate measures to prevent timing attacks.

Keywords: timing attack, cryptanalysis, RSA, Diffie-Hellman, DSS.

Attaques par «Timing»

FIGURE 1: RSAREF Modular Multiplication Times

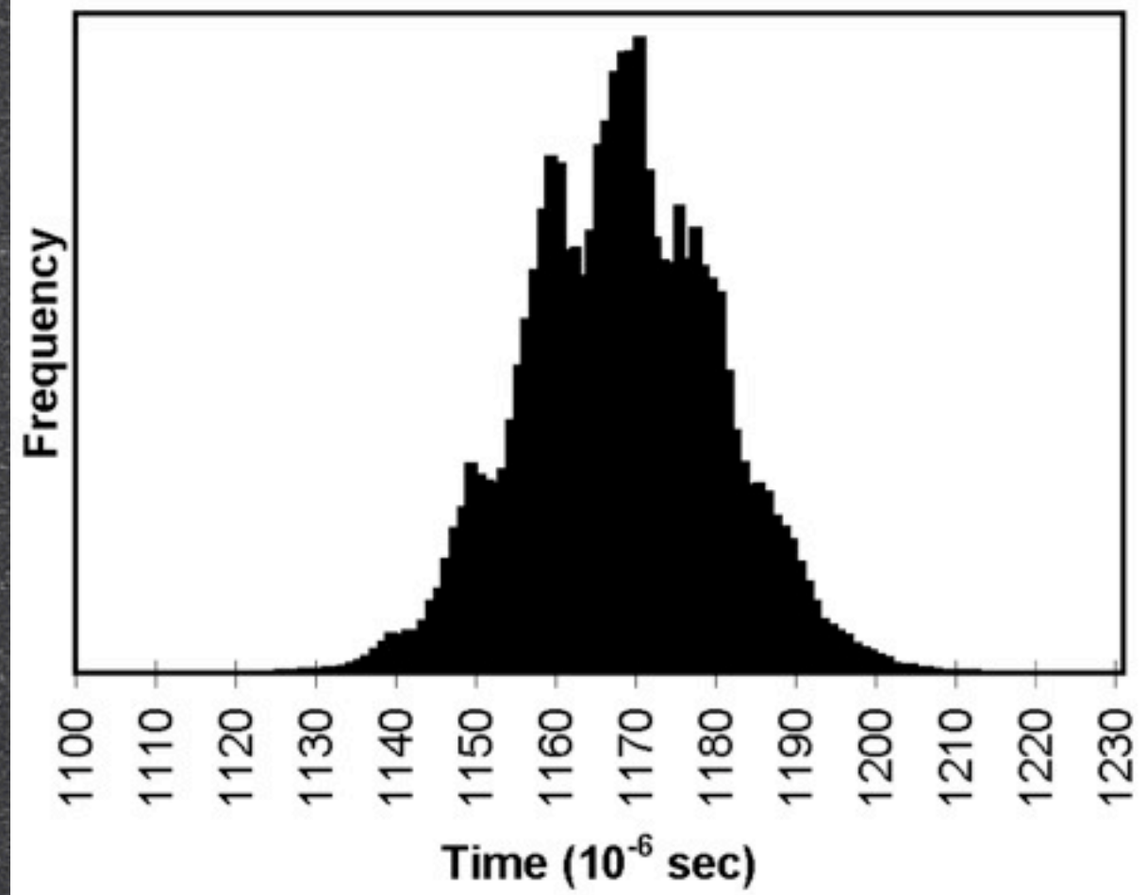
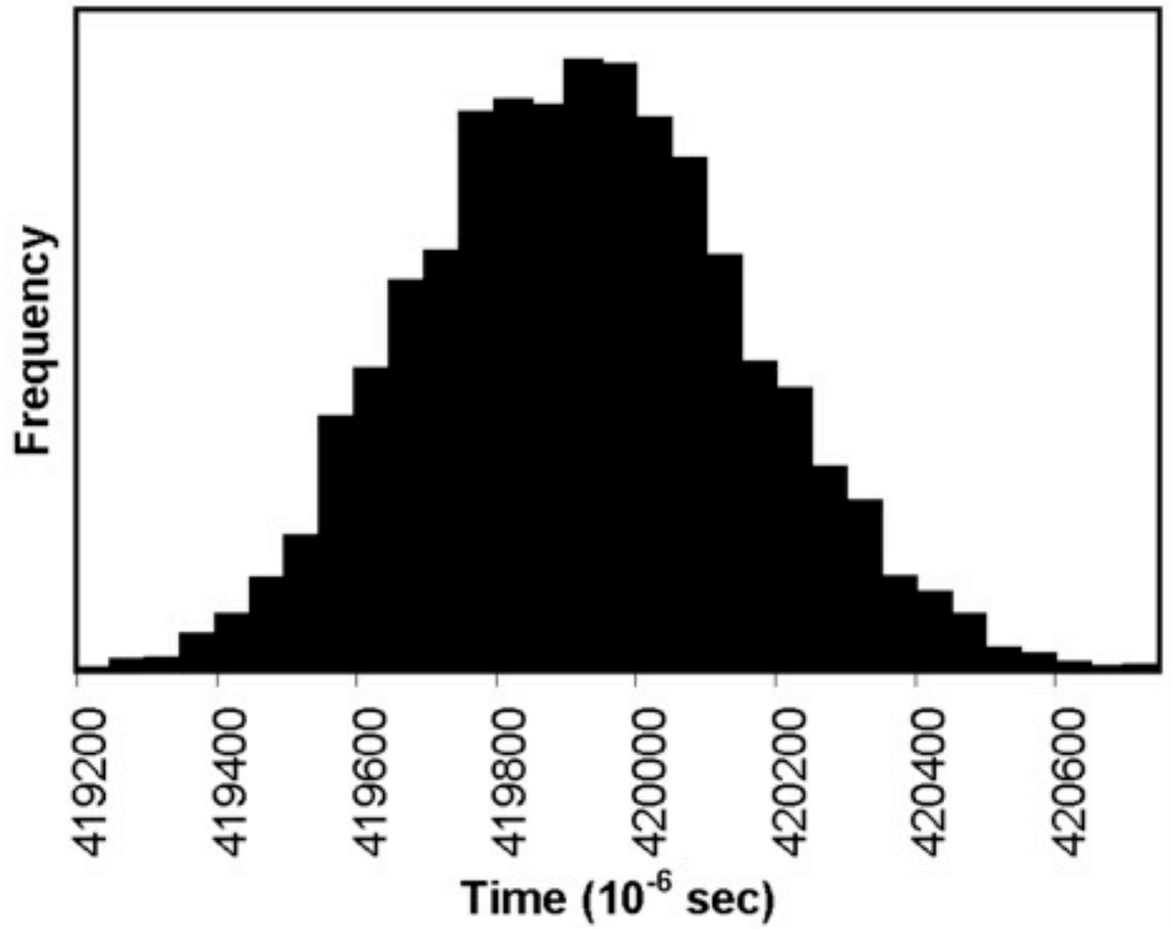
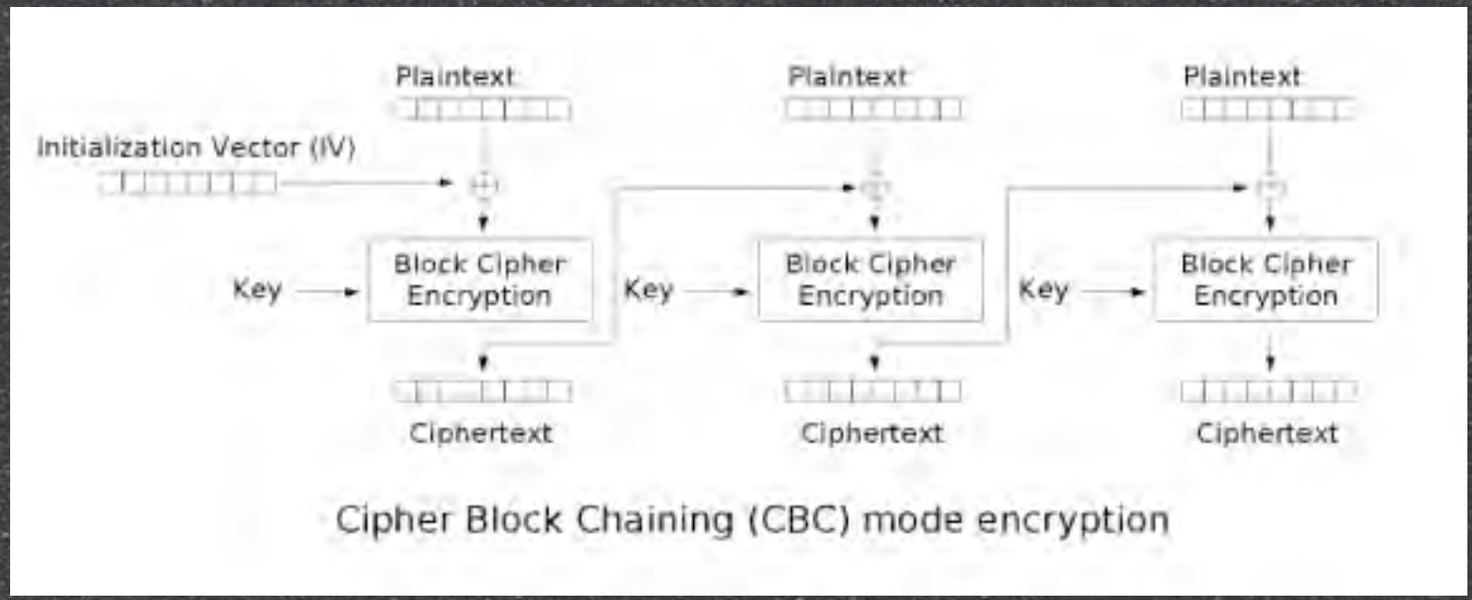


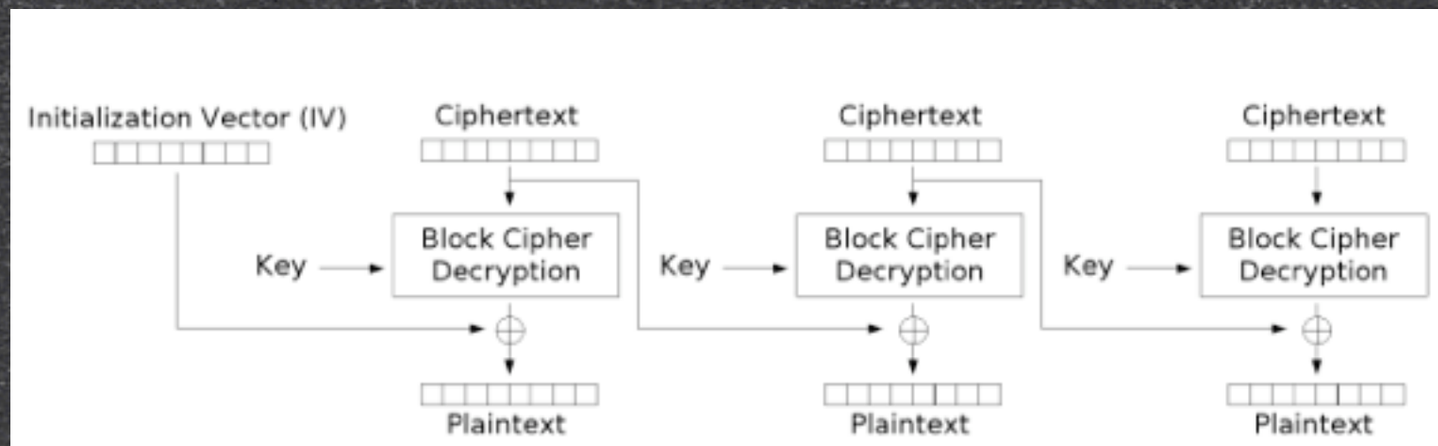
FIGURE 2: RSAREF Modular Exponentiation Times



Attaques par «Timing»



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Attaques par «Timing»

- Le chiffrement en mode CBC exige que les données aient une longueur qui soit multiple de la taille de bloc de l'algorithme de chiffrement utilisé.
  - AES-CBC: multiple of 16 bytes
  - TDES-CBC: multiple of 8 bytes

Attaques par «Timing»

- «Padding» standard avec des blocs de 8 octets:
  - 3 octets manquants: pad avec 03 03 03
  - 7 octets manquants: pad avec 07 07 07 07 07 07 07
  - Pas d'octet manquant: pad avec 08 08 08 08 08 08 08 08

Attaques par «Timing»

- Problème si la routine qui vérifie le «padding» ne travaille pas en temps constant:

Password Interception in a SSL/TLS Channel

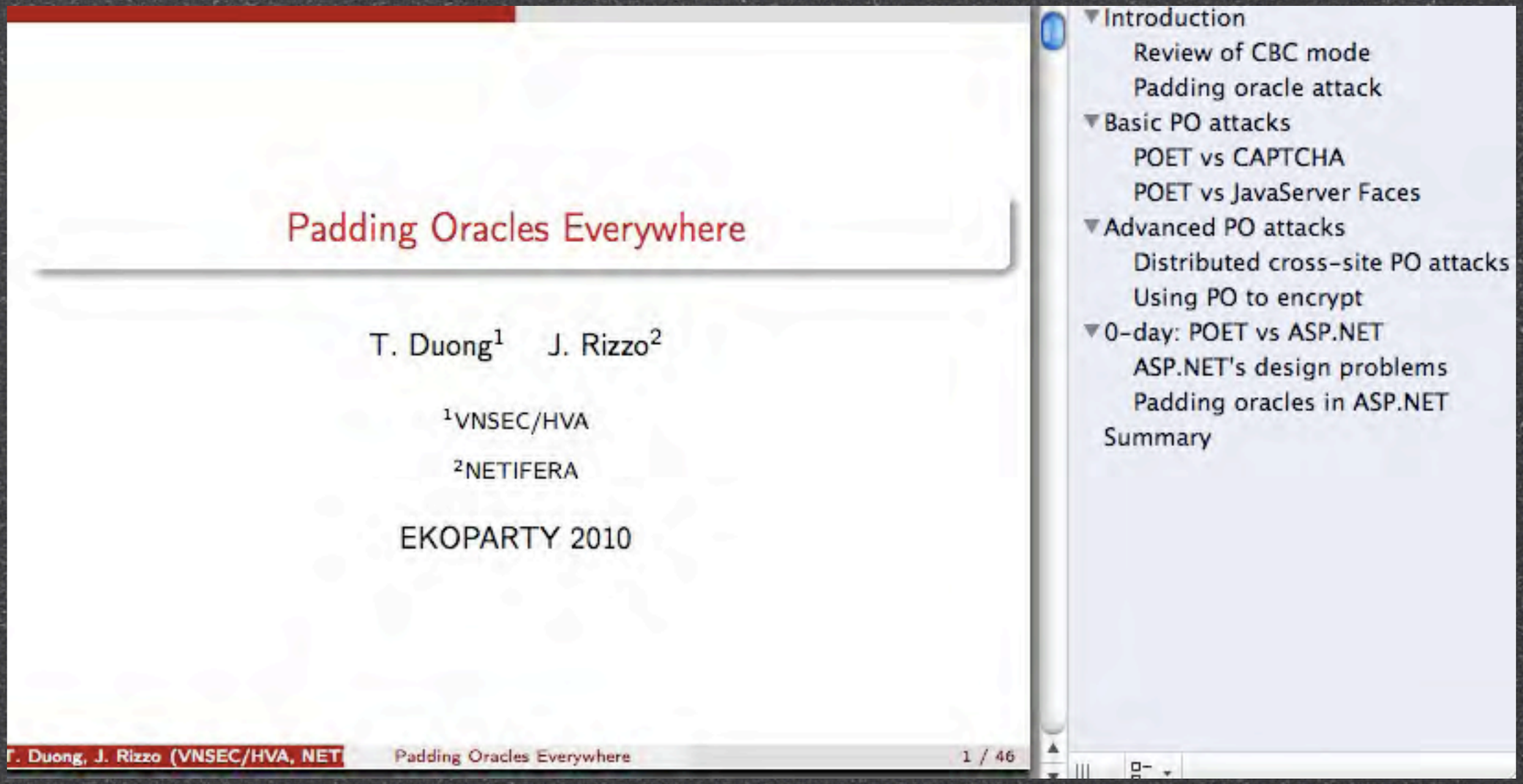
Brice Canvel[1], Alain Hiltgen[2], Serge Vaudenay[1], and Martin Vuagnoux[3]

[1] Swiss Federal Institute of Technology (EPFL) - LASEC
http://lasecwww.epfl.ch
[2] UBS AG
email:alain.hiltgen@ubs.com
[3] EPFL - SSC, and Ilion
http://www.ilionsecurity.ch

Abstract. Simple password authentication is often used e.g. from an email software application to a remote IMAP server. This is frequently done in a protected peer-to-peer tunnel, e.g. by SSL/TLS.
At Eurocrypt'02, Vaudenay presented vulnerabilities in padding schemes used for block ciphers in CBC mode. He used a side channel, namely error information in the padding verification. This attack was not possible against SSL/TLS due to both unavailability of the side channel (errors are encrypted) and premature abortion of the session in case of errors. In this paper we extend the attack and optimize it. We show it is actually applicable against latest and most popular implementations of SSL/TLS (at the time this paper was written) for password interception.
We demonstrate that a password for an IMAP account can be intercepted when the attacker is not too far from the server in less than an hour in a typical setting.
We conclude that these versions of the SSL/TLS implementations are not secure when used with block ciphers in CBC mode and propose ways to strengthen them. We also propose to update the standard protocol.

Attaques par «Timing»

- Autre exemple d'exploitation de problèmes de «padding» :



Attaques par «Timing»

- Attaque contre les caches:

Cache Attacks and Countermeasures: the Case of AES

(Extended Version)

revised 2005-11-20

Dag Arne Osvik[1], Adi Shamir[2] and Eran Tromer[2]

[1] dag.arne@osvik.no

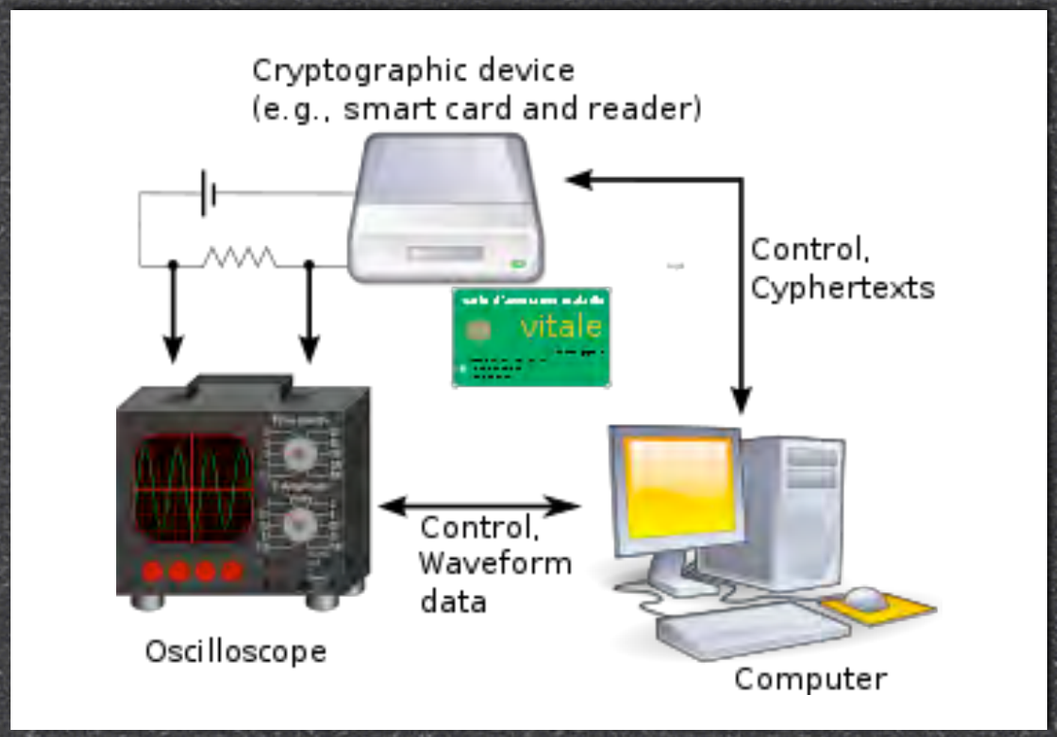
[2] Department of Computer Science and Applied Mathematics,
Weizmann Institute of Science, Rehovot 76100, Israel
{adi.shamir, eran.tromer}@weizmann.ac.il

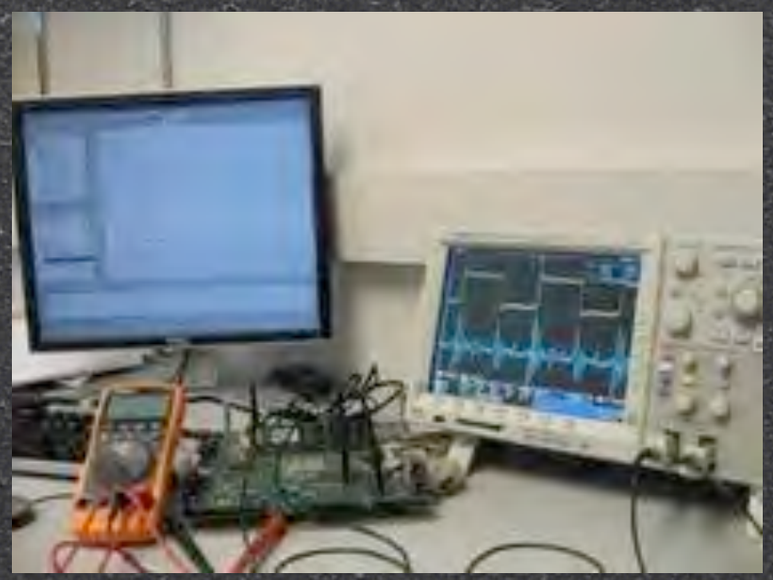
Abstract. We describe several software side-channel attacks based on inter-process leakage through the state of the CPU's memory cache. This leakage reveals memory access patterns, which can be used for cryptanalysis of cryptographic primitives that employ data-dependent table lookups. The attacks allow an unprivileged process to attack other processes running in parallel on the same processor, despite partitioning methods such as memory protection, sandboxing and virtualization. Some of our methods require only the ability to trigger services that perform encryption or MAC using the unknown key, such as encrypted disk partitions or secure network links. Moreover, we demonstrate an extremely strong type of attack, which requires knowledge of neither the specific plaintexts nor ciphertexts, and works by merely monitoring the effect of the cryptographic process on the cache. We discuss in detail several such attacks on AES, and experimentally demonstrate their applicability to real systems, such as OpenSSL and Linux's `dm-crypt` encrypted partitions (in the latter case, the full key can be recovered after just 800 writes to the partition, taking 65 milliseconds). Finally, we describe several countermeasures which can be used to mitigate such attacks.

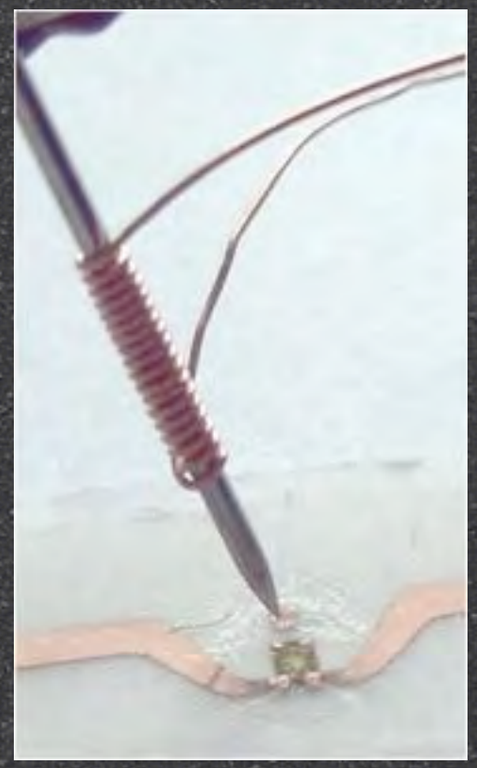
Attaques exploitant les Fuites Physiques

- N’importe quel calcul va inévitablement consommer de l’énergie...
- Si cette consommation d’énergie est corrélée à des valeurs secrètes, alors les secrets sont exposés...

Attaques exploitant les Fuites Physiques







Attaques exploitant les Fuites Physiques

Differential Power Analysis

Paul Kocher, Joshua Jaffe, and Benjamin Jun

Cryptography Research, Inc.
607 Market Street, 5th Floor
San Francisco, CA 94105, USA.
http://www.cryptography.com
E-mail: {paul,josh,ben}@cryptography.com.

Abstract. Cryptosystem designers frequently assume that secrets will be manipulated in closed, reliable computing environments. Unfortunately, actual computers and microchips leak information about the operations they process. This paper examines specific methods for analyzing power consumption measurements to find secret keys from tamper resistant devices. We also discuss approaches for building cryptosystems that can operate securely in existing hardware that leaks information.

Keywords: differential power analysis, DPA, SPA, cryptanalysis, DES

Attaques exploitant les Fuites Physiques

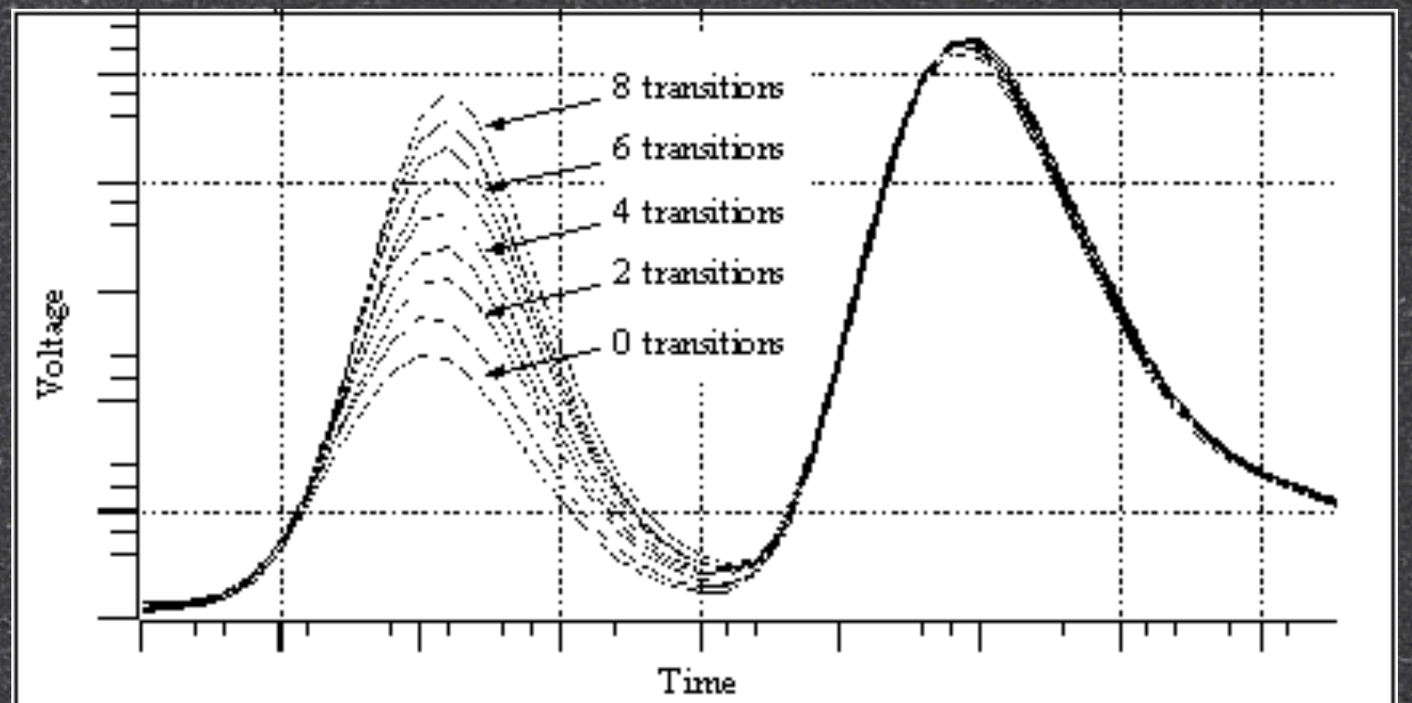


FIGURE 2. Number of Bit Transitions versus Power Consumption.
These results show how the data effects the power levels. The nine overlayed waveforms correspond to the power traces of different data being accessed by an LDA instruction. These results were obtained by averaging the power signals across 500 samples in order to reduce the noise content. The difference in voltage between i transitions and $i+1$ transitions is about 6.5 mV.

Attaques exploitant des Fautes

- Voici un petit bout de code qui pourrait être utilisé dans le cadre de l'authentification de code («secure boot»):

```
if (RSA_verify (signature) ==
RSA_VALID_SIGNATURE) {

  // Perform some critical operation
  } else {
    return NOT_AUTHENTICATED
  }
```

Attaques exploitant des Fautes

- Le même code sous forme de langage machine:

```
...
cmp     $0x0, %ebx
jne     0x64FE89A1
...
```

Toute la sécurité du mécanisme de vérification de la signature RSA repose sur le fait que cette instruction est exécutée...

Attaques exploitant des Fautes

Design Principles for Tamper-Resistant Smartcard Processors

Oliver Kömmerling — *Advanced Digital Security Research, Mühlstraße 7, 66484 Riedelberg, Germany*, ok@adsr.de

Markus G. Kuhn — *University of Cambridge Computer Laboratory, Pembroke Street, Cambridge CB2 3QG, United Kingdom*, mgk25@cl.cam.ac.uk

2.2.1 Glitch Attacks

In a glitch attack, we deliberately generate a malfunction that causes one or more flipflops to adopt the wrong state. The aim is usually to replace a single critical machine instruction with an almost arbitrary other one. Glitches can also aim to corrupt data values as they are transferred between registers and memory. Of the many fault-induction attack techniques on smartcards that have been discussed in the recent literature [11, 12, 16, 17, 18], it has been our experience that glitch attacks are the ones most useful in practical attacks.

We are currently aware of three techniques for creating fairly reliable malfunctions that affect only a very small number of machine cycles in smartcard processors: clock signal transients, power supply transients, and external electrical field transients.

Particularly interesting instructions that an attacker might want to replace with glitches are conditional jumps or the test instructions preceding them. They create a window of vulnerability in the processing stages of many security applications that often allows us to bypass sophisticated cryptographic barriers by simply preventing the execution of the code that detects that an authentication attempt was unsuccessful. Instruction glitches can also be used to extend the runtime of loops, for instance in serial port output routines to see more of the memory after the output buffer [12], or also to reduce the runtime of loops, for instance to transform an iterated cipher function into an easy to break single-round variant [11].

Attaques exploitant des Fautes

OpenSSL et Consoeurs

Pascal Junod -- La Cryptographie

Insomni'hack 2011, 4 mars 2011, Genève, Suisse

OpenSSL et Consoeurs

- De nombreuses librairies cryptographiques open-source et généralistes existent (liste non-exhaustive):
  - OpenSSL
  - libgcrypt
  - Mozilla NSS
  - libtomcrypt
  - NaCl
  - Botan
  - Crypto++
  - cryptlib

OpenSSL et Consoeurs

- Question que j'aimerais traiter (partiellement) maintenant :
- Les librairies cryptographiques généralistes open-source sont-elles sûres ?

OpenSSL et Consoeurs

- Qu'entend-on par «sûres» ?
  - Résistance aux attaques cryptographiques connues
  - Résistance aux attaques «side-channel»
  - (Respect des principes de programmation sécurisée)
  - (Temps de réponse des développeurs face à une faille)
  - ...

Attaque de Manger

- Publiée par James Manger à Crypto'01
- S'applique aux mauvaises implémentations du mécanisme de «padding» de RSA-OAEP
- Transforme une implémentation «mauvaise» en oracle de déchiffrement
- Requière environ 1024 requêtes choisies adaptativement pour décrypter un texte chiffré avec RSA-OAEP 1024 bits.

Attaque de Manger

- Seule information requise: «Est-ce que l'octet le plus significatif du texte clair correspondant à ce texte chiffré est égal à 0x00 ?»

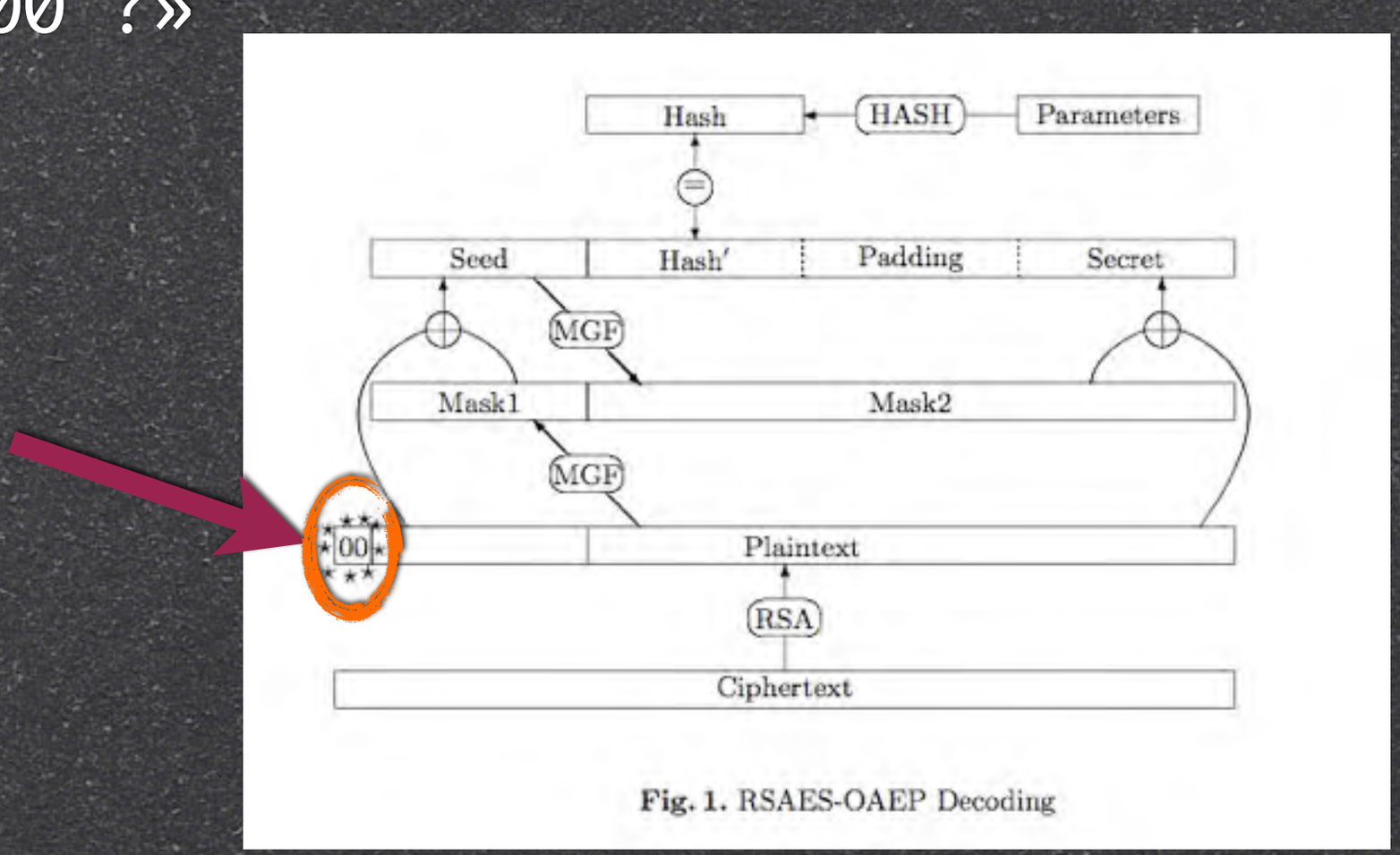


Fig. 1. RSAES-OAEP Decoding

Attaque de Manger

- On peut obtenir cette information (au minimum) grâce:
  - à des messages d’erreur;
  - des différences de «timing».

Attaque de Manger

- Si on jette un coup d’oeil à l’implémentation d’OpenSSL:

```
CHANGES

*) Improve RSA_padding_check_PKCS1_OAEP() check
again to avoid 'wristwatch attack' using huge
encoding parameters (cf. James H. Manger's CRYPTO
2001 paper). Note that the RSA_PKCS1_OAEP_PADDING
case of RSA_private_decrypt() does not use encoding
parameters and hence was not vulnerable.[Bodo
Moeller]
```

Attaque de Manger

- Plus loin:

```
/* crypto/rsa/rsa_oaep.c */
...
/* signalling this error immediately after detection
 * might allow for side-channel attacks (e.g. timing
 * if 'plen' is huge -- cf. James H. Manger, "A
 * Chosen Ciphertext Attack on RSA Optimal
 * Asymmetric Encryption Padding (OAEP) [...]",
 * CRYPTO 2001), so we use a 'bad' flag */
```

Attaque de Manger

- Mais...

```
if (lzero < 0)
                    {
                /* signalling this error immediately after
                 * detection might allow
                 * for side-channel attacks (e.g. timing if
                 * 'plen' is huge
                 * -- cf. James H. Manger, "A Chosen
                 * Ciphertext Attack on RSA Optimal
                 * Asymmetric Encryption Padding (OAEP)
                 * [...]", CRYPTO 2001),
                 * so we use a 'bad' flag */
               bad = 1;
               lzero = 0;
               flen = num; /* don't overflow the memcpy to
                            *padded_from */
               }
```

Attaque de Manger

- Tiré de la page Web de NaCl's:

> The CPU's instruction pointer, branch predictor, etc. are not designed to keep information secret. For performance reasons this situation is unlikely to change. The literature has many examples of successful timing attacks that extracted secret keys from these parts of the CPU.

Attaque de Manger

- Est-ce que c’est temps constant
- DEMO

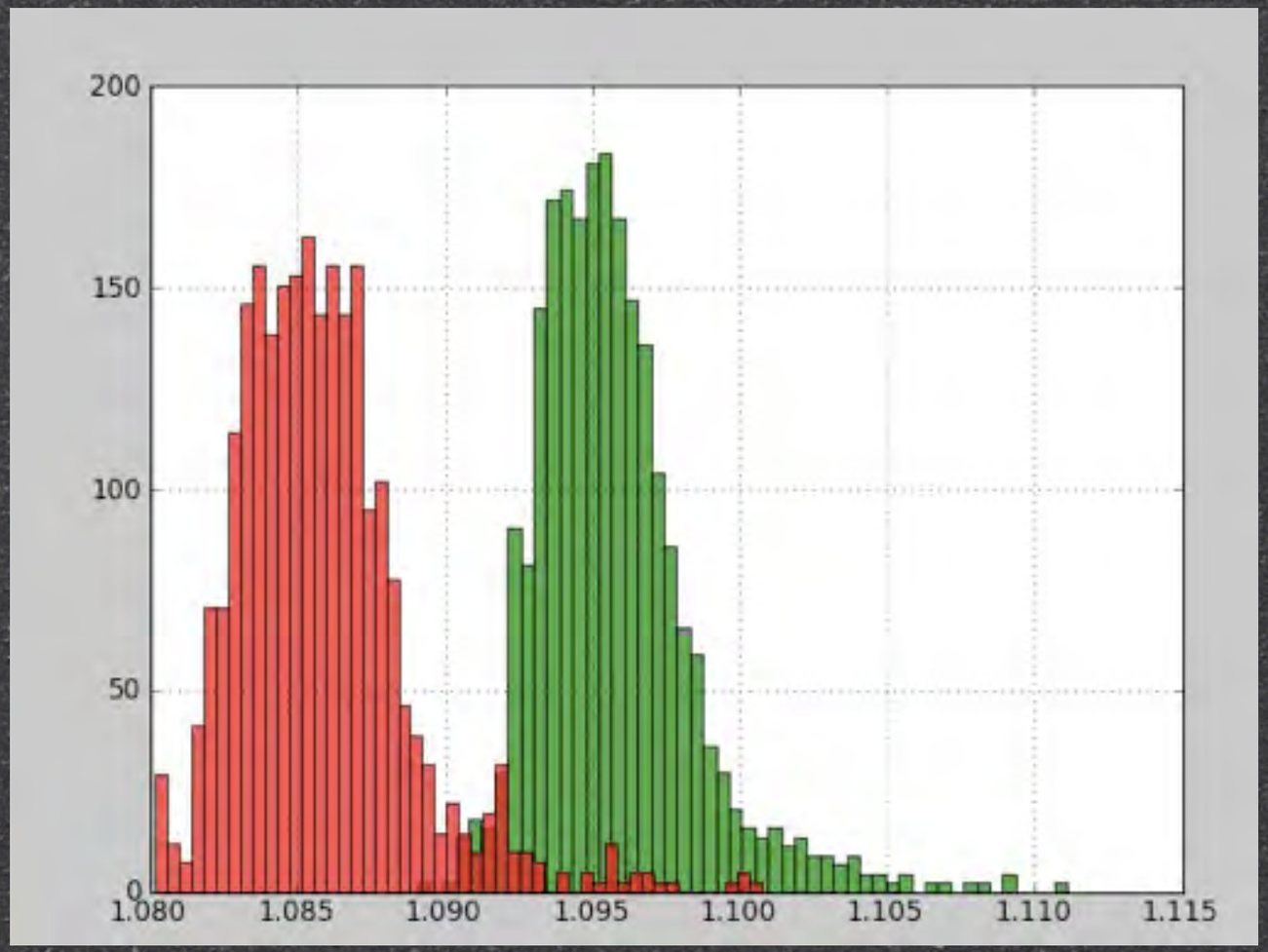
```
macbook-pro-de-pascal-junod:openssl_manger pjunod$ ./junk

[VALID PADDING (20971520)   ] : 10.943075 seconds for
1048576 OAEP check

[INVALID PADDING (-1048576) ] : 10.835983 seconds for
1048576 OAEP checks
```

Attaque de Manger

- Distribution de 1000 mesures indépendantes de 104’858 vérifications



Attaque de Manger

- Est-ce que OpenSSL est cassée ?
  - Sur un PC/serveur:
    - En théorie, oui !
    - En pratique, le nombre de requêtes nécessaires pour isoler le signal du bruit est probablement trop long (réseau, ...)

Attaque de Manger

- Est-ce que OpenSSL est cassée ?
  - Sur une plate-forme embarquée:
    - OUI, ASSUREMENT !!
      - Des mesures au cycle d'horloge près sont possibles.
      - Si le code est temps-constant, on peut utiliser la trace de courant.

Légende

	Attaque par «timing» classique
	Attaques contre les caches
	Attaque exploitant des oracles
	Attaque exploitant une fuite
	Attaque exploitant des fautes
✔	Soin sérieux
~	Soin prix, mais pas toujours correctement/complètement
✖	Pas de soin du tout

OpenSSL

- OpenSSL (http://www.openssl.org)
  - La libraire crypto la plus répandue, la plus déployée

- Possède une excellente réputation

				
✔	~	~	✖	✖

libgcrypt

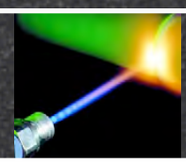
- libgcrypt (http://www.gnupg.org)
  - Maintenue par les développeurs de GnuPG

				
✔	✖	✖	✖	✖

libtomcrypt

- libtomcrypt (http://www.libtom.org)
- Ecrite par un ado fou de crypto

- Principale argument: rapidité...

				
✖	✖	✖	✖	✖

Mozilla NSS

- NSS (http://www.mozilla.org)
- Maintenue par la fodation Mozilla

				
✔	✖	~	✖	~

NaCl

- NaCl (http://nacl.cace-project.eu)
  - Ecrite par des cryptologues dans le cadre du projet européen CACE
- Arguments
  - Implémentations très rapides
  - Pas de branchement dépendant de données secrètes

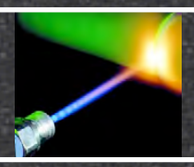
NaCl

- Malheureusement, n’implémente principalement que des primitives exotiques.

				
✔	✔	✔	✖	✖

Botan

- Botan (http://botan.randombit.net)
  - Ecrite en C++

				
✔	✔	~	✖	~

Botan

```
// Is this vulnerable to timing attacks?
   for(u32bit i = HASH_LENGTH + Phash.size(); i !=
tmp.size(); ++i)
{
if(tmp[i] && !delim_idx)
   {
   if(tmp[i] == 0x01)
      delim_idx = i;
   else
      delim_ok = false;
   }
}
```

Crypto++

- Crypto++ (http://www.cryptopp.com)
  - Projet maintenu par Wei Dai
  - Certifiée FIPS 140-2 level 1

				
✔	~	~	✖	✖

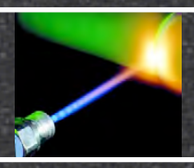
Crypto++

```
bool invalid = false;

// convert from bit length to byte length
if (oaepBlockLen % 8 != 0)
    {
        invalid = (oaepBlock[0] != 0) ||invalid;
        oaepBlock++;
    }
```

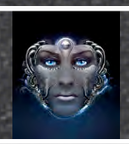
cryptlib

- cryptlib (http://www.cs.auckland.ac.nz/~pgut001/cryptlib)
- Ecrite par Peter Gutman

				
✔	✖	~	~	~

Conclusion

Conclusion

					
OpenSSL	✔	~	~	✖	✖
libgcrypt	✔	✖	✖	✖	✖
libtomcrypt	✖	✖	✖	✖	✖
NSS	✔	✖	~	✖	~
NaCl	✔	✔	✔	✖	✖
Botan	✔	✔	~	✖	~
Crypto++	✔	~	~	✖	✖
cryptlib	✔	✖	~	~	~

Conclusion

- Bien que l'on dispose aujourd'hui de bonnes librairies cryptographiques open-source, elles ne sont pas, ou pas complètement protégées contre
  - les attaques par oracle,
  - les attaques par fuite,
  - et les attaques par faute.

Conclusion

- Cela ne pose pas (trop) de problème si on les utilise sur un PC/serveur standard.
- Mais les utiliser sur du matériel embarqué est TRES RISQUE !

Conclusion

- On attend toujours une librairie cryptographique généraliste et open-source correctement sécurisée !

Information de Contact

- Site Web http://crypto.junod.info
- Twitter @cryptopathe
- E-mail pascal@junod.info