Cryptographic Backdooring

JP Aumasson

KUDELSKI SECURITY

/me: @veorq http://aumasson.jp

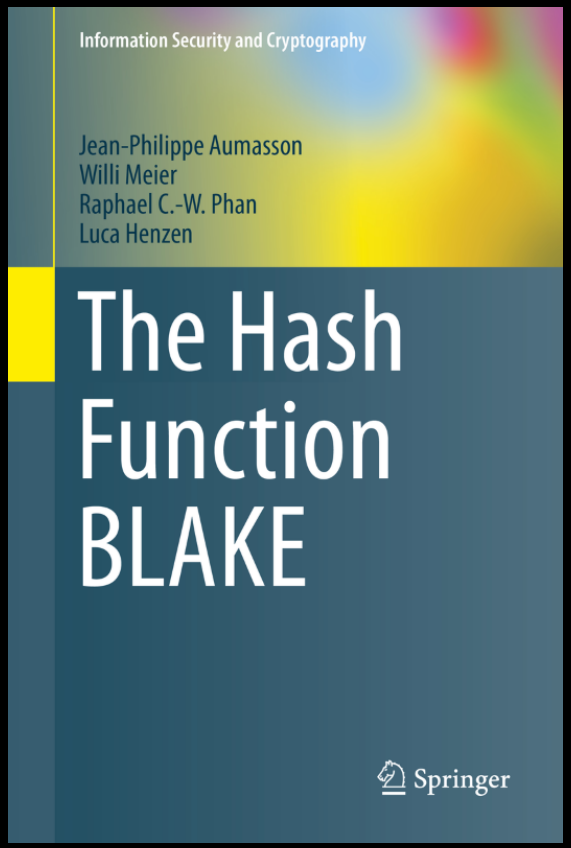
BLAKE(2), SipHash, NORX

https://password-hashing.net

https://cryptocoding.net

https://malicioussha1.github.io

DahuCon



Agenda

Why this talk?
Backdooring 101
Sabotage tactics
A perfect backdoor
Conclusion

Why this talk?

You may not be interested in backdoors,
but backdoors are interested in you

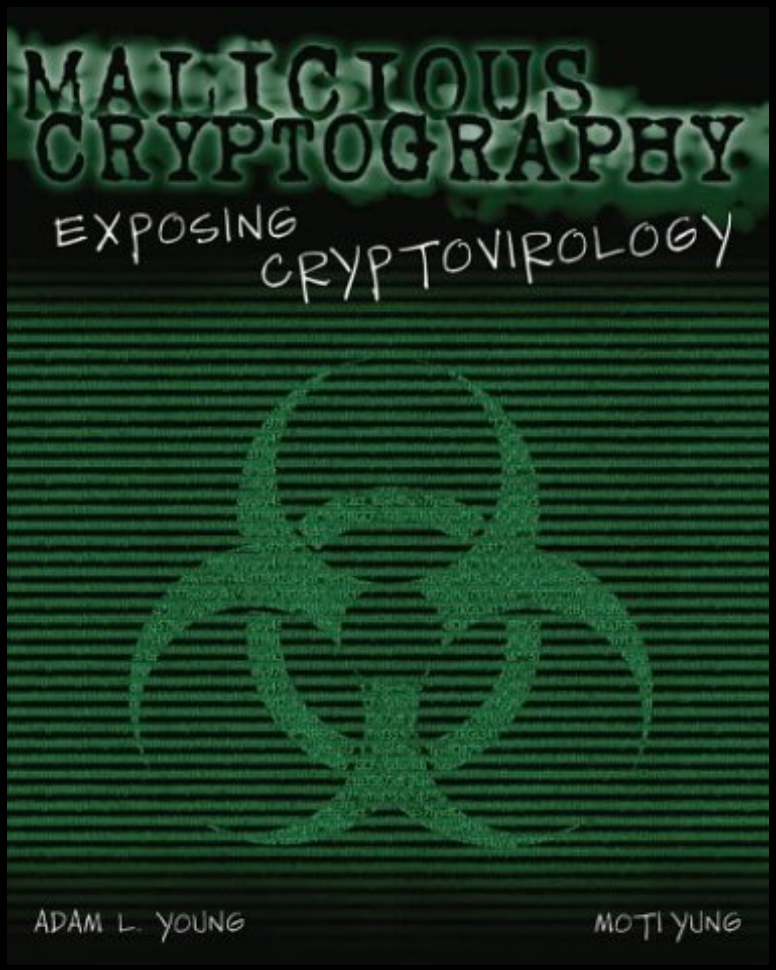
(U) Base resources in this project are used to:

- (TS//SI//REL TO USA, FVEY) Insert vulnerabilities into commercial encryption systems, IT systems, networks, and endpoint communications devices used by targets.
- (TS//SI//REL TO USA, FVEY) Collect target network data and metadata via cooperative network carriers and/or increased control over core networks.
- (TS//SI//REL TO USA, FVEY) Leverage commercial capabilities to remotely deliver or receive information to and from target endpoints.
- (TS//SI//REL TO USA, FVEY) Exploit foreign trusted computing platforms and technologies.
- (TS//SI//REL TO USA, FVEY) Influence policies, standards and specification for commercial public key technologies.
- (TS//SI//REL TO USA, FVEY) Make specific and aggressive investments to facilitate the development of a robust exploitation capability against Next-Generation Wireless (NGW) communications.
- (U//FOUO) Maintain understanding of commercial business and technology trends.

NSA's BULLRUN program

Public research mostly inexistant

2004


MALICIOUS CRYPTOGRAPHY
EXPOSING CRYPTOVIROLOGY
ADAM L. YOUNG
MOTI YUNG

Surreptitiously Weakening Cryptographic Systems

Bruce Schneier[1] Matthew Fredrikson[2] Tadayoshi Kohno[3] Thomas Ristenpart[2]

[1] *Co3 Systems* [2] *University of Wisconsin* [3] *University of Washington*

February 9, 2015

Abstract

Revelations over the past couple of years highlight the importance of understanding malicious and surreptitious weakening of cryptographic systems. We provide an overview of this domain, using a number of historical examples to drive development of a weaknesses taxonomy. This allows comparing different approaches to sabotage. We categorize a broader set of potential avenues for weakening systems using this taxonomy, and discuss what future research is needed to provide sabotage-resilient cryptography.

http://eprint.iacr.org/2015/097.pdf

Bad reputation: surveillance, deception

“a back door for the government can easily —and quietly—become a back door for criminals and foreign intelligence services.”

Security “Front Doors” vs. “Back Doors”: A Distinction Without a Difference

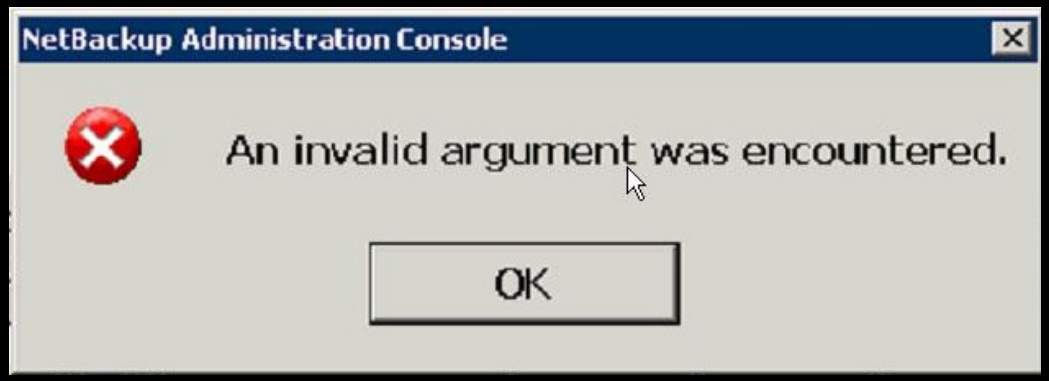
By *Jeffrey Vagle* and *Matt Blaze*

Friday, October 17, 2014 at 2:06 PM

http://justsecurity.org/16503/security-front-doors-vs-back-doors-distinction-difference/

And terrorists etc.

(Like internet and encryption)



“It increases the ‘attack surface’ of the system, providing new points of leverage that a nefarious attacker can exploit.”


Security “Front Doors” vs. “Back Doors”: A Distinction Without a Difference

By *Jeffrey Vagle* and *Matt Blaze*

Friday, October 17, 2014 at 2:06 PM

http://justsecurity.org/16503/security-front-doors-vs-back-doors-distinction-difference/

matt blaze
@mattblaze

Following

Crypto backdoors are dangerous even if you trust the government not to abuse them. We simply don't know how to build them reliably.

Not well understood, by the public

Especially **crypto** backdoors

Why doing research about backdoors?

Detect backdoors

If you have to implement a backdoor, whatever the reasons, better do it well

Backdooring 101

What’s a backdoor?

Not a trapdoor

(Covert rather than overt)

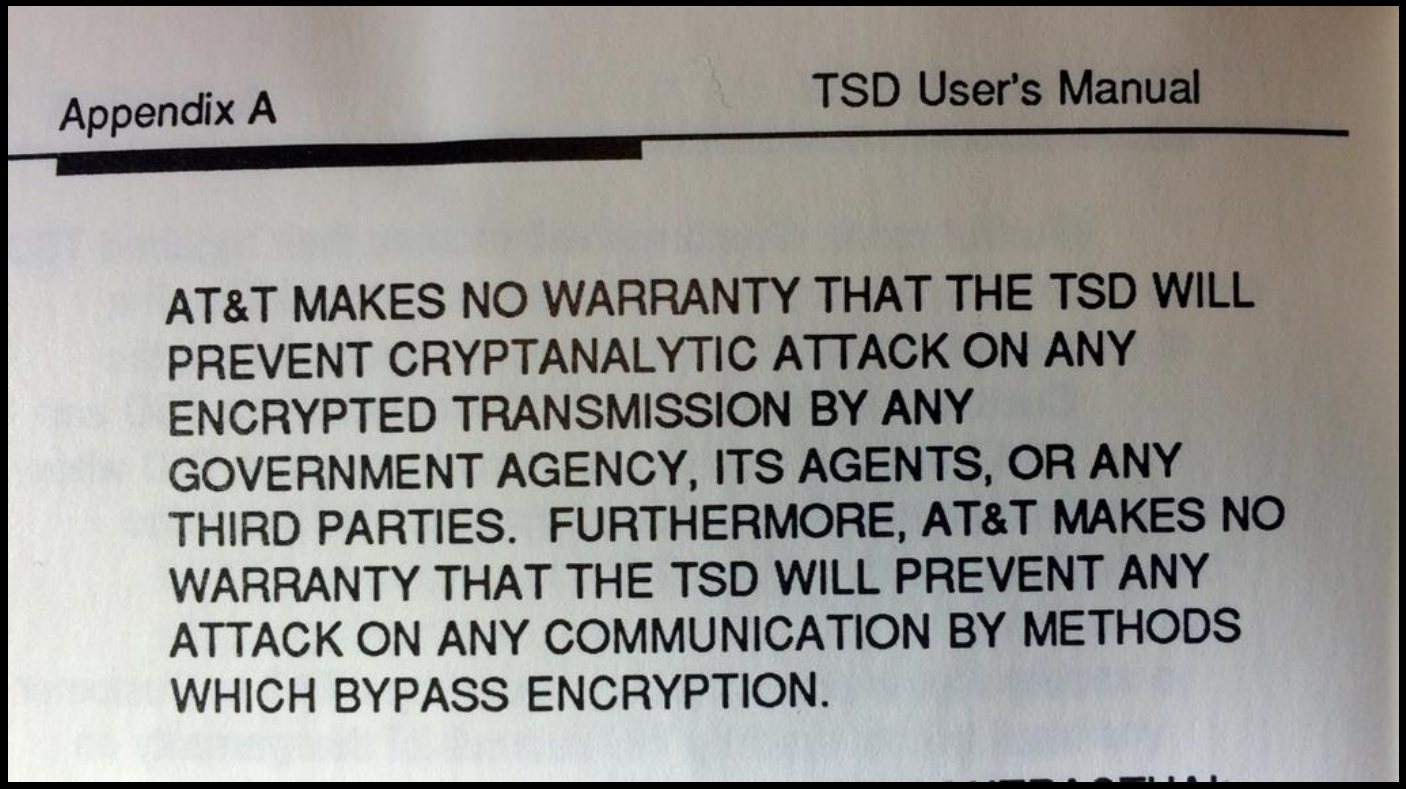
“A feature or defect that allows surreptitious access to data”

Weakened algorithms
(A5/2, GMR, etc.)

Covert channels

(Exfiltration of keys, etc.)

Key escrow



Appendix A TSD User's Manual

AT&T MAKES NO WARRANTY THAT THE TSD WILL PREVENT CRYPTANALYTIC ATTACK ON ANY ENCRYPTED TRANSMISSION BY ANY GOVERNMENT AGENCY, ITS AGENTS, OR ANY THIRD PARTIES. FURTHERMORE, AT&T MAKES NO WARRANTY THAT THE TSD WILL PREVENT ANY ATTACK ON ANY COMMUNICATION BY METHODS WHICH BYPASS ENCRYPTION.

Clipper chip phone AT&T TSD3600

“An undocumented way to get access to a computer system or the data it contains”

Breakthrough silicon scanning discovers backdoor in military chip (DRAFT of 05 March 2012)

Sergei Skorobogatov
University of Cambridge
Cambridge, UK
sps32@cam.ac.uk

Christopher Woods
Quo Vadis Labs
London, UK
chris@quovadislabs.com

Bugdoors

Backdoors that look like bugs

What’s a good backdoor?

Undetectable

Observables look legit

Requires non-trivial RE

Deniable

Looks unintentional
Isn’t incriminating

NOBUS (no one but us)

Exploitation requires a **secret**:
Keys, algorithm, protocol, etc.

Can also be specific privilege, skill, etc.

Reusable

Multiple times, against multiple targets

Usable without being revealed

(Unlike Flame's MD5 collision)

Unmalleable

Not easily tweaked to be exploited by another party
Difficult to replicate without all details

Forward-secure

If the backdoor is detected,
previous exploits aren’t compromised

Simple

Minimize code, logic, memory,etc.

Sabotage tactics

Constants

Choose constants that allow you to compromise the security

Malicious Hashing: Eve's Variant of SHA-1

Ange Albertini[1], Jean-Philippe Aumasson[2], Maria Eichlseder[3], Florian Mendel[3], and Martin Schläffer[3]

40 bits modified

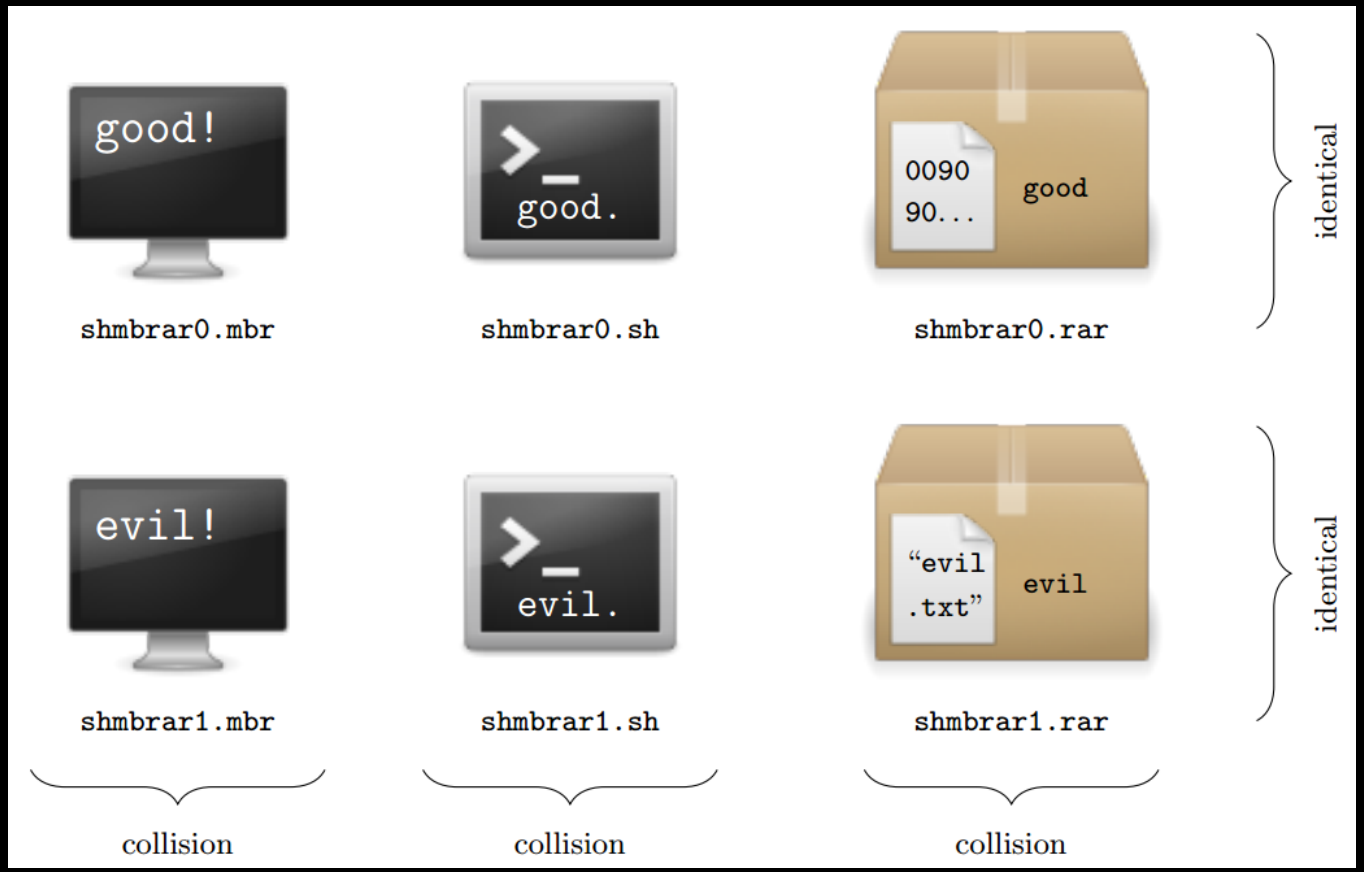
Colliding binaries, images, archives

Full control on the content, NOBUS

(BSidesLV/DEFCON/SAC 2014)

https://malicioussha1.github.io

2 distinct files, 3 valid file formats



NIST curves' coefficients

Hashes of unexplained 16-byte seeds, e.g.

c49d3608 86e70493 6a6678e1 139d26b7 819f7e90

(Speculation, not evidence of backdoor)

Notion of **rigidity**

Or suspiciousness of the constants:

“a feature of a curve-generation process, limiting the number of curves that can be generated”

http://safecurves.cr.yp.to/rigid.html

Curve25519	fully rigid ✓	Prime chosen "as close as possible to a power of 2" for efficiency reasons ("save time in field operations"). Prime chosen "slightly below 32k bits, for some k" for efficiency reasons ("no serious concerns regarding wasted space"). k=8 chosen for "a comfortable security level". 2^255-19 chosen above 2^255+95, 2^255-31, 2^254+79, 2^253+51, 2^253+39 "because 19 is smaller than 31, 39, 51, 79, 95". Montgomery curve shape y^2=x^3+Ax^2+x chosen for efficiency ("to allow extremely fast x-coordinate point operations"). (A-2)/4 selected as a small integer for efficiency ("to speed up the multiplication by (A-2)/4"). Curve and twist orders required to be {4*prime,8*prime} for security ("protect against various attacks ... here 4, 8 are minimal"). Primes required to be above 2^252 for security ("theoretical possibility of a user's secret key matching the prime"), ruling out A=358990 and A=464586. A=486662 chosen as smallest positive integer meeting these requirements.
BN(2,254)	fully rigid ✓	p chosen sparse, close to 2^256, within BN family; using u=−(2^62 + 2^55 + 1). p congruent 3 modulo 4 to have z^2+1 irreducible; b=2 to have twist be y^2=x^3+ (1 − 2i).
brainpoolP256t1	somewhat rigid ✓	Several unexplained decisions: Why SHA-1 instead of, e.g., RIPEMD-160 or SHA-256? Why use 160 bits of hash input independently of the curve size? Why pi and e instead of, e.g., sqrt(2) and sqrt(3)? Why handle separate key sizes by more digits of pi and e instead of hash derivation? Why counter mode instead of, e.g., OFB? Why use overlapping counters for A and B (producing the repeated 26DC5C6CE94A4B44F330B5D9)? Why not derive separate seeds for A and B?
ANSSI FRP256v1	**trivially manipulatable**	No explanation provided.
NIST P-256	**manipulatable**	Coefficients generated by hashing the unexplained seed c49d3608 86e70493 6a6678e1 139d26b7 819f7e90.
secp256k1	somewhat rigid ✓	GLV curve with 256 bits and prime order group; prime and coefficients not fully explained but might be minimal
E-382	fully rigid ✓	
M-383	fully rigid ✓	
Curve383187	fully rigid ✓	p is largest prime smaller than 2^383; B=1; A > 2 is as small as possible.
brainpoolP384t1	somewhat rigid ✓	See brainpoolP256t1.
NIST P-384	**manipulatable**	Coefficients generated by hashing the unexplained seed a335926a a319a27a 1d00896a 6773a482 7acdac73.

How to manipulate curve standards: a white paper for the black hat

Daniel J. Bernstein[1,2], Tung Chou[1], Chitchanok Chuengsatiansup[1], Andreas Hülsing[1], Tanja Lange[1], Ruben Niederhagen[1], and Christine van Vredendaal[1]

"The BADA55-VPR curves illustrate the fact that 'verifiably pseudorandom' curves with 'systematic' seeds generated from 'nothing-up-my-sleeve numbers' also do not stop the attacker from generating a curve with a one-in-a-million weakness."

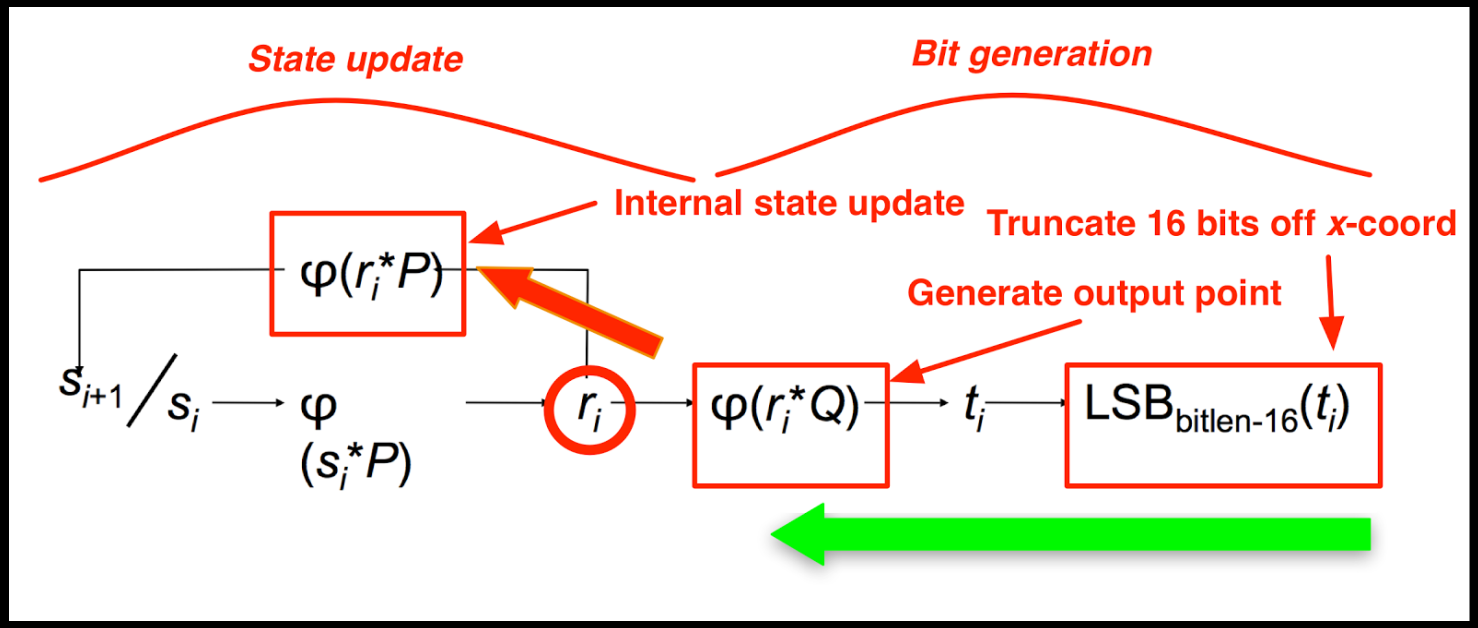
http://safecurves.cr.yp.to/bada55.html

This program can generate millions of plausible values for “somewhat rigid” constants

https://github.com/veorq/NUMSgen

Is it possible to find many “fully rigid” designs?

Dual_EC_DRBG
(NSA design, NIST standard)



http://blog.cryptographyengineering.com/2013/09/the-many-flaws-of-dualecdrbg.html

If n such that nQ = P is known, RNG is broken (NOBUS)

Constants are anything that is.. constant
Arithmetic operations, S-boxes, etc.

A backdoor in AES?

The Rijndael algorithm was adopted as Advanced Encryption Standard (AES) by U. S. NIST in 2001 in FIPS-197 [6]. AES is currently widely deployed around the world and frequently used by unsuspecting users. In this note we show that a key component of AES in fact contains a backdoor the allows the Belgian Government and The Catholic Church (the forces behind Rijndael / AES design, who obviously hid the backdoor in the cipher) to secretly eavesdrop on all AES communications. This is why the National Security Agency has been actively promoting the use of AES in public networks [1, 5, 8, 10].

Fig. 1. This paper describes an extremely efficient potential algebraic attack against the U. S. Advanced Encryption Standard (AES).

(Research article by the honorable Dr. Gavekort: https://mjos.fi/doc/gavekort_kale.pdf)

Sabotaged AES S-box??

3 Security Analysis

We see that the algebraic degree of the underlying transform is $\deg S' = -1$. This degree is very low, in fact lower than zero, the degree recommended by the Chinese Government ("sinkhole transform"). For reference, the U.S. National Security Agency recommends degree 1 ("the identity transform") for all but the most confidential data. AES has been clearly designed to offer even lower security than these proposals against Algebraic Attacks of Courtois [3].

Bruce Schneier (in joint work with Euclid) has developed an algorithm to compute multiplicative inverses in rings mod n, even when the factorization of n is not known [11]. We see that $e = -1$ is clearly unsuitable for modern cryptography [11]. We call this the Euclid-Courtois-Gavekort-Schneier (ECGS) Algebraic Attack on AES. Based to extrapolations from reduced versions, we estimate that attack complexity against AES-128 is $2^{127.884763735192088017115417615704834017 88}$ with 2^{127} precomputation.

AES S-box is just the inverse $x \rightarrow x^{-1}$ in $GF(2^8)$!

A better S-box for AES!

However, finite fields of characteristic two (such as our $\mathsf{GF}(2^8)$) have an unique $\sqrt{x}$ for each x, including -1. Since $\sqrt{-1}$ is clearly defined, we call the resulting S-Box an **Irrational Permutation** (IP).

5 Improved High-Degree AES Variant KALE

Figure 4 shows the S-Box used by KALE. The S-Box is the only difference between KALE and AES. Appendix A gives a full trace of KALE128 execution that can be used to verify implementation correctness. There are no other modifications to the Key Schedule, number of rounds, etc. Note that the very first elements are unchanged since zero mapped to zero in the AES inversion and $\sqrt{0} = 0$, and furthermore $1^{-1} = \sqrt{1}$. The same masking constant `0x63` is used.

The algebraic degree of $\sqrt{x}$ in real and complex fields is $\frac{1}{2}$, but in a multiplicative subgroup of finite field of size $2n$ it is actually n. Therefore the degree is actually 128. We may write interchangeably $\sqrt{x} = x^{128}$. The cycling properties are also greatly improved for S'.

Can you find the *real* backdoor?

Key generation

Make session keys predictable

3G/4G AKA

Session keys = hash(master key, **rand**)

Delegate tactical intercepts with low-entropy **rand** values

Precompute and share session keys

(Just a possibility, not making allegations)

Hide weak parameters

RSA

Hide small public exponent

with some tricks to avoid detection

and recover using Boneh-Durfee-Frankel result

Simple Backdoors for RSA Key Generation

Claude Crépeau[1] and Alain Slakmon[2]

(CT-RSA 2003)

Key generation as a covert channel for itself

RSA

Hide bits of prime factors in *n*

Recover using Coppersmith's method

Similar to "Pretty-Awful-Privacy" (Young-Yung)

Simple Backdoors for RSA Key Generation

Claude Crépeau[1] and Alain Slakmon[2]

(CT-RSA 2003)

Lesson: don’t outsource keygen

Implementations

Slightly deviate from the specs
Omit some verifications
etc.

Small subgroup attacks

Omit (EC)DH pubkey validation

A Key Recovery Attack on Discrete Log-based Schemes Using a Prime Order Subgroup*

Chae Hoon Lim[1] and Pil Joong Lee[2]

(CRYPTO 1997)

Validation of Elliptic Curve Public Keys

Adrian Antipa[1], Daniel Brown[1], Alfred Menezes[2], René Struik[1], and Scott Vanstone[2]

(PKC 2003)

TLS MitM

Incomplete cert verification

“Misuse”

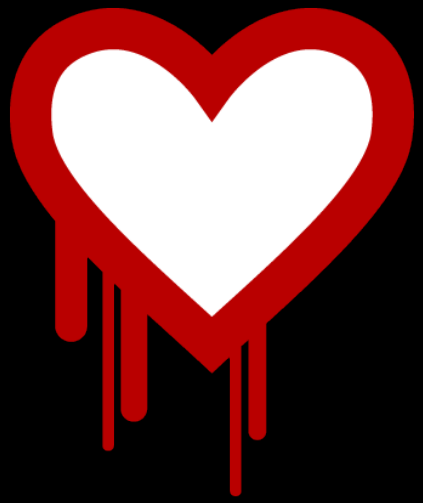
Repeated stream cipher nonces

NOBUS unlikely...

Software

Bugdoors in the crypto

Deniability may be plausible



```
goto fail;
goto fail;

goto cleanup;
```

Probably unintentional

Not NOBUS anyway

RC4 bugdoor (Wagner/Biondi)

```
#define TOBYTE(x) (x) & 255
#define SWAP(x,y) do { x^=y; y^=x; x^=y; } while (0)

static unsigned char A[256];
static int i=0, j=0;

unsigned char encrypt_one_byte(unsigned char c) {
    int k;
    i = TOBYTE( i+1 );
    j = TOBYTE( j + A[i] );
    SWAP( A[i], A[j] );
    k = TOBYTE( A[i] + A[j] );
    return c ^ A[k];
}
```

RC4 bugdoor (Wagner/Biondi)

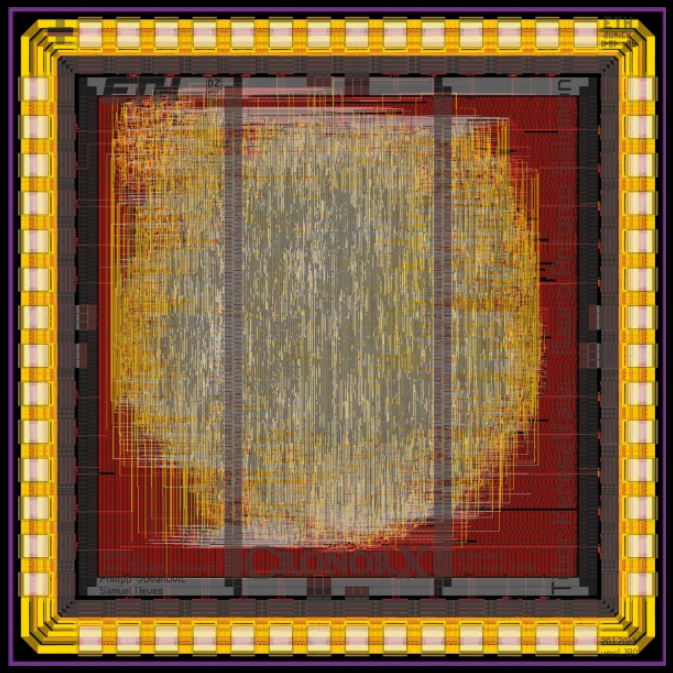
```
#define TOBYTE(x) (x) & 255
#define SWAP(x,y) do { x^=y; y^=x; x^=y; } while (0)

static unsigned char A[256];
static int i=0, j=0;

unsigned char encrypt_one_byte(unsigned char c) {
    int k;
    i = TOBYTE( i+1 );
    j = TOBYTE( j + A[i] );
    SWAP( A[i], A[j] );  /* what if ( i == j ) ?*/
    k = TOBYTE( A[i] + A[j] );
    return c ^ A[k];
}
```

Hardware

IC trojans



Malicious modification of a chip

At design (HDL), fab (netlist), distribution (IC)

Detection difficult

"Undetectable by *optical* RE!"

Stealthy Dopant-Level Hardware Trojans *

Georg T. Becker[1], Francesco Regazzoni[2], Christof Paar[1,3], and Wayne P. Burleson[1]

(CHES 2013)

"Maybe, but not with *electronic* imaging (SEM)"

Reversing Stealthy Dopant-Level Circuits

Takeshi Sugawara[1], Daisuke Suzuki[1], Ryoichi Fujii[1], Shigeaki Tawa[1]
Ryohei Hori[2], Mitsuru Shiozaki[2], and Takeshi Fujino[2]

(CHES 2014)

Bug Attacks

Eli Biham[1], Yaniv Carmeli[1], and Adi Shamir[2]

CPU multiplier X × Y = Z correct
except for one “magic” pair (X, Y)

Exploitable to break RSA, ECC, etc.

2^{128} pairs for 64-bit MUL, detection unlikely

A perfect backdoor

http://phili89.wordpress.com/2010/05/24/the-perfect-crime-project-38/

Covert channel with a malicious RNG

NOBUS thanks public-key encryption

Undetectable thanks to proven indistinguishability

Compute **X** = Enc(pubkey, *secret data to exfiltrate*)

X values should look random

Use **X** as IVs for AES-CBC

Public-key encryption scheme with ciphertexts indistinguishable from random strings?

Elligator: Elliptic-curve points indistinguishable from uniform random strings

Daniel J. Bernstein[1,4]
djb@cr.yp.to

Mike Hamburg[2]
mhamburg@cryptography.com

Anna Krasnova[3]
anna@mechanical-mind.org

Tanja Lange[4]
tanja@hyperelliptic.org

Elligator curves

E-382	True ✓	Elligator 2: Yes.
M-383	True ✓	Elligator 2: Yes.
Curve383187	True ✓	Elligator 2: Yes.
brainpoolP384t1	**False**	Elligator 2: No.
NIST P-384	**False**	Elligator 2: No.
Curve41417	True ✓	Elligator 2: Yes.
Ed448-Goldilocks	True ✓	Elligator 2: Yes.
M-511	True ✓	Elligator 2: Yes.
E-521	True ✓	Elligator 2: Yes.

http://safecurves.cr.yp.to/ind.html

RNG circuit must be hidden

For example in FPGA/PLD, difficult to RE

Communications and computations

Indistinguishable from those of a clean system

In case of **full RE**

Backdoor detected but unexploitable,
Previous covert coms remain safe (FS)

What can be exfiltrated? **RNG state**

Can give past and future session keys, depending on the RNG construction

Many other techniques…

Conclusion

All this is quite basic

(Credit: @krypt3ia)

And that’s only for crypto

Should we really worry about backdoors?

Or first fix bugs and usability issues?

“Competition to write or modify crypto code that appears to be secure, but actually does something evil”

https://underhandedcrypto.com/

16 submissions received

Winner: John Meacham

sabotaged AES, confusion in standard type redefinition

Runner-up: Gaëtan Leurent

ZK identification protocol, buggy Hamming weight

Thank you!