



Norwegian University of
Science and Technology

LEGACY CRYPTO AND ATTACKS ON TLS

TTM4205 – Lecture 6

Tjerand Silde

01.09.2026

Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

Backdoors

Summary

Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

Backdoors

Summary

Reference Material

These slides are based on:

- ▶ The referred papers in the slides
- ▶ JPA: parts of chapters 5, 6, and 13
- ▶ DW: parts of chapters 9 and 16

Legacy Crypto is...

- ▶ Old and outdated crypto
- ▶ Insecure, weakened, or flawed crypto
- ▶ Crypto regulated by export control
- ▶ Potentially backdoored crypto
- ▶ Key escrow and surveillance
- ▶ Downgradable crypto protocols



Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

Backdoors

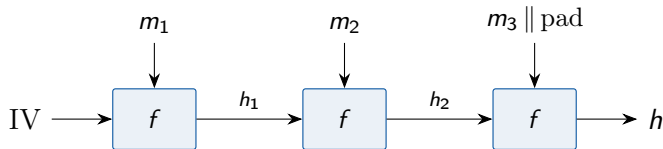
Summary

Ciphers of the 1990s

Several newer ciphers developed in the 1990s and 2000s were the leading standards for many years, and they are still used in various protocols and products used on or connected to the Internet today.

There has been a variety of attacks, and here are some examples...

Merkle-Damgård Hashing



MD5 and SHA-1 are both built this way: pad the message, split it into blocks, and chain a compression function f through them, with $h_i = f(h_{i-1}, m_i)$.

If two messages collide, then so does anything appended to them, which is what turns a collision into a forged signature, MAC, or certificate.

MD5 Hash Function

- ▶ Hash function outputting 128 bits
- ▶ Designed by Ron Rivest in 1991
- ▶ First identical-prefix collisions found in 2004
- ▶ First chosen-prefix collisions found in 2007
- ▶ Used to create a rogue CA certificate in 2008, and by the Flame malware to forge a Microsoft signature in 2012
- ▶ Revoked in most (!) applications by 2014 (!)

How MD5 Collisions Are Found

Wang and Yu (2005) work with *modular differences*: they fix a difference Δ between two messages and follow how it propagates through the 64 steps.

Two blocks. The first block introduces a difference in the chaining value; the second is chosen so that the difference cancels, leaving $h_2 = h'_2$.

Message modification. The path only works if the internal state satisfies hundreds of conditions. The early rounds depend on the message directly, so the message words are adjusted to force those conditions.

Then brute force: about 2^{39} compressions in the original paper, hours back then, milliseconds today. Chosen-prefix collisions add a birthday phase first, to bring two arbitrary prefixes into a difference the path can cancel.

Year	Identical-prefix collision cost	Chosen-prefix collision cost
< 2004	2^{64} generic	2^{64} generic
2004	2^{40} [WY05]	—
2005	2^{37} [Kli05]	—
2006	2^{32} [Kli06, Ste06]	2^{49} [SLdW07c]
2007	2^{25} [Ste07]	—
2008	2^{21} [XLF08]	—
2009	2^{16} [SSA ⁺ 09]	2^{39} [SSA ⁺ 09]
2020	2^{16} [SSA ⁺ 09]	2^{39} [SSA ⁺ 09]

Figure: <https://www.marc-stevens.nl/research/papers/CC21Chapter-S.pdf>

Why This Breaks RADIUS

RADIUS authenticates the server's answer with a plain MD5 hash: the Response Authenticator is H over the response, the request authenticator, and the shared secret. There is no HMAC and, by default, no transport security.

The attack. A man in the middle between the access point and the RADIUS server injects an attribute into the *request*, chosen so that the honest Access-Reject and the forged Access-Accept collide under MD5.

The server hashes its Reject and that is a valid authenticator for the attacker's Accept. The shared secret is never needed, and the attacker is logged in.

Blast-RADIUS (CVE-2024-3596, 2024) made this practical by computing the chosen-prefix collision fast enough to fit inside the login timeout. The fix is to require the Message-Authenticator attribute, an HMAC, or to run it over TLS.



RADIUS/UDP vulnerable to improved MD5 collision attack

2024-07-09



Sharon Goldberg



Miro Haller (Guest Author)



Nadia Heninger (Guest Author)



Michael Milano (Guest Author)



Dan Shumow (Guest Author)



Marc Stevens (Guest Author)



Adam Suhl (Guest Author)

Figure: <https://blog.cloudflare.com/radius-udp-vulnerable-md5-attack>

RADIUS/UDP Considered Harmful

Sharon Goldberg
Cloudflare

Miro Haller
UC San Diego

Nadia Heninger
UC San Diego

Mike Milano
BastionZero

Dan Shumow
Microsoft Research

Marc Stevens
Centrum Wiskunde & Informatica

Adam Suhl
UC San Diego

Figure: <https://www.blastradius.fail/pdf/radius.pdf>

SHA-1 Hash Function

- ▶ Hash function outputting 160 bits
- ▶ Designed by the NSA in 1995, after SHA-0 in 1993
- ▶ First identical-prefix collision in 2017
- ▶ First chosen-prefix collision in 2020
- ▶ Revoked in most (!) applications by 2020 (!)

How SHA-1 Collisions Are Found

SHA-1 expands 16 message words into 80 linearly, so a single bit flipped in the expansion propagates in a predictable pattern: a *disturbance vector*.

Around each disturbance one builds a *local collision*, a small set of corrections that cancels it a few steps later. Chaining these gives a differential path over two near-collision blocks whose difference cancels at the end.

SHAttered (2017) turned it into an identical-prefix collision at a cost of $2^{63.1}$ compressions: two PDF files with the same digest.

Shambles (2020) reached a *chosen-prefix* collision at $2^{63.4}$, for roughly \$45,000 of rented GPU time, and used it to forge a PGP certification. At that price, "expensive" is no longer a defence.

SHA-1

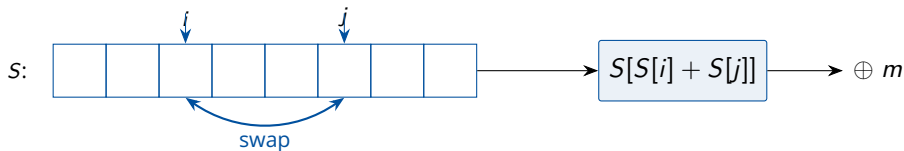
Year	Identical-prefix collision cost		Chosen-prefix collision cost	
< 2005	2^{80}	generic	2^{80}	generic
2005	2^{69}	[WYY05b]	—	
	$(u : 2^{63}$	[WYY05a])	—	
2007	$(u : 2^{61}$	[MRR07])	—	
2009	$(w : 2^{52}$	[MHP09])	—	
2013	2^{61}	[Ste13b]	2^{77}	[Ste13b]
2017	G : $2^{63.1}$	[SBK⁺17]	—	
2019	—		$G : 2^{67}$	[LP19]
2020	$2^{61} / G : 2^{61.2}$ [Ste13b] / [LP20]		G : $2^{63.4}$ [LP20]	

Figure: <https://www.marc-stevens.nl/research/papers/CC21Chapter-S.pdf>

RC4 Encryption Scheme

- ▶ Symmetric stream cipher using at least 40-bit keys
- ▶ Designed by Ron Rivest in 1987, made public in 1994
- ▶ Used in WEP (1997), WPA (2003), and SSL/TLS (1995)
- ▶ Biased from the very first bytes: the second output byte is zero twice as often as it should be
- ▶ Fluhrer-Mantin-Shamir key recovery in 2001, WEP broken in 2004
- ▶ Prohibited in TLS by RFC 7465 in 2015

RC4: The Keystream



The state is a permutation S of the 256 byte values and two indices. The key schedule starts from the identity permutation and swaps entries according to the key; each step then advances i , adds $S[i]$ to j , swaps those two entries, and outputs the entry at the summed index.

Twenty lines of code, which is why it spread. The trouble is that the first bytes of the keystream are not uniform.

Weaknesses in the Key Scheduling Algorithm of RC4

Scott Fluhrer¹, Itsik Mantin², and Adi Shamir²

¹ Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134
`sfluhrer@cisco.com`

² Computer Science department, The Weizmann Institute, Rehovot 76100, Israel.
`{itsik,shamir}@wisdom.weizmann.ac.il`

Figure: https://www.mattblaze.org/papers/others/rc4_ksaproc.pdf

Why RC4 Fails in Practice

A stream cipher is a one-time pad with a generated pad, $c = m \oplus \text{KS}(k)$, so everything rests on never generating the same pad twice.

Keystream reuse. Two messages under one key give $c \oplus c' = m \oplus m'$: the key drops out, and known protocol headers reveal the keystream for everything else sent under it.

Related keys. WEP prepends a 24-bit IV to the shared key, so the key schedule is rerun on keys differing in three bytes. Fluhrer, Mantin and Shamir turned that into key recovery in 2001, and WEP fell in practice by 2004.

3DES Encryption Scheme

- ▶ DES: a 16-round Feistel cipher with a 56-bit key and a 64-bit block, standardized by NIST as FIPS 46 in 1977
- ▶ Exhaustive key search became practical: the EFF machine found a key in days in 1998
- ▶ 3DES: encrypt, decrypt, encrypt with three keys, standardized in ANSI X9.52 (1998) and NIST SP 800-67
- ▶ Meet-in-the-middle brings the 168 key bits down to 112 bits of security
- ▶ The block size is the real problem: 64 bits means collisions after 2^{32} blocks, which is the Sweet32 attack of 2016
- ▶ Deprecated by NIST and disallowed for encryption after 2023

How DES and 3DES Were Broken

Cryptanalysis did not do it. Differential cryptanalysis needs 2^{47} chosen plaintexts and linear cryptanalysis 2^{43} known ones, neither of which is available in practice. The S-boxes were designed to resist the first, twenty years before it was published in the open literature.

The key was simply too short. The EFF built Deep Crack in 1998 for about \$250,000: 1856 custom chips searching the 2^{56} keys, first in 56 hours, and together with distributed.net in 22 hours in 1999.

3DES answers the key length, not the block size. Meet-in-the-middle leaves 112 bits of security, which is fine, but the block is still 64 bits.

Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

Backdoors

Summary

Legacy Public Key Crypto

While we have attacks against symmetric key ciphers making them obsolete, we do not have groundbreaking attacks against legacy public key ciphers.

However, we need to be careful when setting parameters and composing different schemes in more complex protocols.

Here are some examples...

Weak DH

- ▶ Improved discrete log $\rightarrow$ Must use larger keys
- ▶ Working in all of $\mathbb{Z}_p^*$ $\rightarrow$ Leaks the Legendre symbol of m
 - ▶ the order $p - 1$ is composite, and Pohlig-Hellman reduces the discrete log to its largest prime factor $q \mid (p - 1)$
 - ▶ squares and non-squares are distinguishable, so DDH is false in $\mathbb{Z}_p^*$
 - ▶ work in the prime-order subgroup $\mathbb{G}_q$ instead, with g of order q
- ▶ Supersingular curves $\rightarrow$ Pairings break Decisional DH
- ▶ Choose safe curves? $\rightarrow$ Standardized P-256, X25519, ...

Curves That Are Too Friendly

A curve is only as hard as the field it can be mapped into. The Weil and Tate pairings send a curve point pair to $\mathbb{F}_p^k$, where k is the *embedding degree*.

For a supersingular curve $k \leq 6$, so the pairing maps the ECDLP into a field small enough for index calculus. That is the MOV attack: an elliptic curve problem we cannot solve becomes a finite field problem we can.

Given $Q = x \cdot P$, compute $\alpha = e(P, P)$ and $\beta = e(P, Q) = \alpha^x$, then solve for x in $\mathbb{F}_p^k$ instead. This is the weekly problem *MOVing Problems*.

Ordinary curves have k of the size of n , which makes the mapped problem far harder than the original. Anomalous curves fail differently, so parameter validation is part of using elliptic curves at all.

Elliptic Curve Cryptography in Practice

Joppe W. Bos¹, J. Alex Halderman², Nadia Heninger³, Jonathan Moore, Michael Naehrig¹,
and Eric Wustrow²

¹ Microsoft Research

² University of Michigan

³ University of Pennsylvania

Figure: <https://eprint.iacr.org/2013/734.pdf>

Weak RSA

- ▶ Improved factoring $\rightarrow$ Must use larger keys
- ▶ If $d < \frac{1}{3}N^{1/4} \rightarrow$ Wiener's attack to recover d
- ▶ Small e and the same m sent to e parties $\rightarrow$ Håstad's attack
- ▶ Several attacks based on Coppersmith's method
 - ▶ Message recovery for a stereotyped m with a small unknown part
 - ▶ Message recovery against short padding, and related messages
 - ▶ Factoring n given half of the bits of p or q
- ▶ If $e < \sqrt{N}$, given $\frac{1}{4} \log_2 N$ bits of $d \rightarrow$ Can reconstruct d
- ▶ PKCS #1 v1.5 padding $\rightarrow$ Bleichenbacher's padding oracle attack

Twenty Years of Attacks on the RSA Cryptosystem

Dan Boneh
dabo@cs.stanford.edu

Figure: <https://crypto.stanford.edu/~dabo/papers/RSA-survey.pdf>

RSA Challenges

Challenge Name	Digits	Bits	Date Factored	Factored by
RSA-100	100	330	Apr 1, 1991	A. K. Lenstra
RSA-110	110	364	Apr 14, 1992	A. K. Lenstra and M.S. Manasse
RSA-120	120	397	Jul 9, 1993	T. Denny et al.
RSA-130	130	430	Apr 10, 1996	A. K. Lenstra et al.
RSA-140	140	463	Feb 2, 1999	H. te Riele et al.
RSA-150	150	496	Apr 16, 2004	K. Aoki et al.
RSA-155	155	512	Aug 22, 1999	H. te Riele et al.
RSA-160	160	530	Apr 1, 2003	J. Franke et al.
RSA-170	170	563	Dec 29, 2009	D. Bonenberger and M. Krone
RSA-576	174	576	Dec 3, 2003	J. Franke et al.
RSA-180	180	596	May 8, 2010	S. A. Danilov and I. A. Popovyan
RSA-190	190	629	Nov 8, 2010	A. Timofeev and I. A. Popovyan
RSA-640	193	640	Nov 2, 2005	J. Franke et al.
RSA-200	200	663	May 9, 2005	J. Franke et al.
RSA-210	210	696	Sep 26, 2013	R. Propper
RSA-704	212	704	Jul 2, 2012	S. Bai, E. Thomé and P. Zimmermann
RSA-220	220	729	May 13, 2016	S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann
RSA-230	230	762	Aug 15, 2018	S. S. Gross
RSA-768	232	768	Dec 12, 2009	T. Kleinjung et al.
RSA-240	240	795	Nov 24, 2019	F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann
RSA-250	250	829	Feb 28, 2020	F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann

Table 1. The solved RSA Challenges

Figure: <https://eprint.iacr.org/2021/894.pdf>



Key Sizes

	Parameter	Legacy	Future System Use	
			Near Term	Long Term
Symmetric Key Size	k	80	128	256
Hash Function Output Size	m	160	256	512
MAC Output Size	m	80	128	256
RSA Problem	$\ell(n) \geq$	1024	3072	15360
Finite Field DLP	$\ell(p^n) \geq$	1024	3072	15360
	$\ell(p), \ell(q) \geq$	160	256	512
ECDLP	$\ell(q) \geq$	160	256	512
Pairing	$\ell(q^n) \geq$	1024	3072	15360
	$\ell(p), \ell(q) \geq$	160	256	512

Table 2. Key Size Analysis, where $\ell(\cdot)$ refers to the bit-length of the parameter.

Figure: <https://eprint.iacr.org/2021/894.pdf>

Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

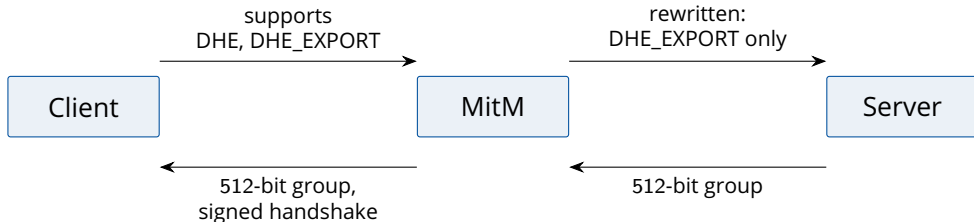
Backdoors

Summary

Static Finite Field DH

- ▶ A MitM attack on TLS ≤ 1.2 can choose weak ciphers
- ▶ Export Diffie-Hellman accepts 512-bit prime groups
- ▶ One week of pre-computation $\rightarrow$ each discrete log takes about a minute
- ▶ The cost sits in the group, not the connection: nearly all servers share a handful of primes
- ▶ 2015: breaking one 1024-bit prime would expose 18% of the top HTTPS domains, a second one 66% of IPsec VPNs and 26% of SSH servers

Logjam: The Downgrade



The server signs the parameters but not their *strength*, so the client cannot tell that it was downgraded. With the group precomputed, the man in the middle solves the discrete log during the handshake and holds the session key.

Why a 512-Bit Group Falls

The number field sieve for discrete logarithms splits in two: a large precomputation that depends only on the prime p , and a much cheaper descent that depends on the target g^x .

That split is what makes shared parameters dangerous. Nearly all export-grade servers used one of a handful of standard primes, so one precomputation buys every connection to all of them, forever.

A 512-bit prime took about a week to precompute in 2015, after which each individual log took roughly a minute: fast enough to run inside a handshake.

What Those Attacks Actually Are

- ▶ **BEAST** (2011): TLS 1.0 chains the IV from the previous record, so it is predictable, and CBC becomes chosen-plaintext distinguishable
- ▶ **CRIME** and **BREACH** (2012–13): compression before encryption, so the ciphertext length leaks whether the secret appears in attacker-chosen data
- ▶ **Lucky 13** (2013): MAC-then-encrypt makes the padding removal visible in the timing, which we return to next lecture
- ▶ **POODLE** (2014): SSL 3.0 padding bytes are not covered by the MAC, so a downgrade to SSL 3.0 revives the padding oracle

What Those Attacks Actually Are

- ▶ **FREAK** (2015): servers accept an export-grade RSA key they were never asked for, and 512-bit RSA factors in hours
- ▶ **Logjam** (2015): the same idea for Diffie-Hellman, with the precomputation amortized over a shared prime
- ▶ **DROWN** (2016): an SSLv2 server sharing the RSA key decrypts *TLS* sessions, so an obsolete protocol elsewhere breaks the current one
- ▶ **Sweet32** (2016) and **ROBOT** (2017): the 64-bit block and Bleichenbacher, both still reachable two decades later

Old Attacks on TLS

RC4

- Roos's Bias 1995
- Fluhrer, Martin & Shamir 2001
- Klein 2005
- Combinatorial Problem 2001
- Royal Holloway 2013
- Bar-mitzvah 2015
- NOMORE 2015

RSA-PKCS#1 v1.5 Encryption

- Bleichenbacher 1998
- Jager 2015
- DROWN 2016

Renegotiation

- Marsh Ray Attack 2009
- Renegotiation DoS 2011
- Triple Handshake 2014

3DES

- Sweet32

AES-CBC

- Vaudenay 2002
- Boneh/Brumley 2003
- BEAST 2011
- Lucky13 2013
- POODLE 2014
- Lucky Microseconds 2015

Compression

- CRIME 2012

MD5 & SHA1

- SLOTH 2016
- SHAttered 2017

Figure: https://owasp.org/www-chapter-london/assets/slides/OWASPLondon20180125_TLSv1.3_Andy_Brodie.pdf

Downgrade Attacks on TLS

TLS: a long year of downgrade attacks

- POODLE TLS 1.2 → SSLv3 [Dec'14]
- FREAK RSA-2048 → RSA-512 [Mar'15]
- **LOGJAM** **DH-2048 → DH-512** **[May'15]**
- BLEICH? RSA-Sign → RSA-Enc [Aug'15]
- **SLOTH** **RSA-SHA256 → RSA-MD5** **[Jan'16]**

Figure: <https://rwc.iacr.org/2016/Slides/Downgrade.pdf>

From TLS 1.2 to 1.3

- ▶ Removed RSA for key exchange
- ▶ Removed RC4, 3DES and Camellia
- ▶ Removed MD5 and SHA-1 hash functions
- ▶ Removed AES-CBC encryption mode
- ▶ Removed static (EC) Diffie-Hellman
- ▶ Only standardized groups/curves

What Is a Cipher Suite?

A TLS 1.2 cipher suite names four choices at once, for instance ECDHE-RSA-AES128-GCM-SHA256:

- ▶ the key exchange (ephemeral elliptic-curve Diffie-Hellman),
- ▶ the authentication (an RSA signature on the handshake),
- ▶ the record encryption (AES-128 in GCM, an AEAD mode),
- ▶ and the hash for the handshake and the key schedule.

Every combination is a suite of its own, which is how TLS 1.2 ended up with hundreds of them, many carrying obsolete algorithms.

New Cipher Suites

TLS 1.3 names only the record algorithms, and negotiates the key exchange and the signature separately. That leaves five suites:

- ▶ TLS_AES_128_GCM_SHA256
- ▶ TLS_AES_256_GCM_SHA384
- ▶ TLS_CHACHA20_POLY1305_SHA256
- ▶ TLS_AES_128_CCM_SHA256
- ▶ TLS_AES_128_CCM_8_SHA256

All five are AEADs, and the key exchange is always ephemeral, so forward secrecy is no longer optional.

Matthew Green's Blog

- ▶ Standards: <https://blog.cryptographyengineering.com/2011/10/04/how-standards-go-wrong-constructive>
- ▶ Logjam: <https://blog.cryptographyengineering.com/2015/05/22/attack-of-week-logjam>
- ▶ FREAK: <https://blog.cryptographyengineering.com/2015/03/03/attack-of-week-freak-or-factoring-nsa>



Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

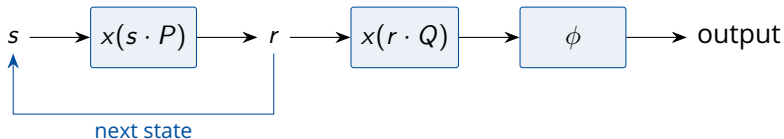
Backdoors

Summary

Dual EC

- ▶ A PRNG designed by the NSA and standardized by NIST in SP 800-90A
- ▶ Its output is argued to be indistinguishable under a DDH-like assumption on elliptic curves
- ▶ The input is a random seed s and two curve points P and Q
- ▶ The standard fixes P and Q without saying where they came from
- ▶ Let $x(P)$ denote the x coordinate of the point P
- ▶ Let ϕ truncate $x(P)$ to a bit string, dropping the top 16 bits

Dual EC: One Step



The truncation ϕ drops only 16 bits, so a few thousand candidates cover the full point $r \cdot Q$. Anyone knowing e with $P = e \cdot Q$ computes $e \cdot (r \cdot Q) = r \cdot P$, whose x coordinate is the next state.

One output block therefore reveals the whole future of the generator, and whoever picked P and Q could have picked e first.

A Security Analysis of the NIST SP 800-90 Elliptic Curve Random Number Generator

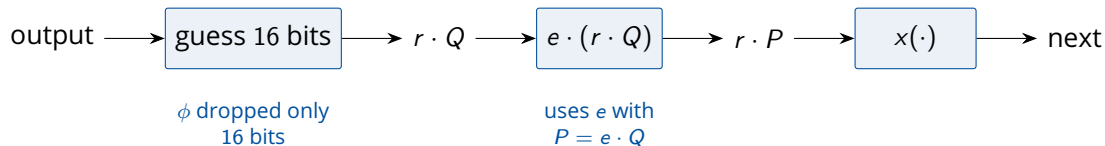
Daniel R. L. Brown* and Kristian Gjøsteen†

February 15, 2007

Figure: <https://eprint.iacr.org/2007/048.pdf>

This is predictable if you know the discrete log relating P and Q

Dual EC: The Backdoor



The generator is defined with P and Q ; the backdoor is the scalar e between them. Nothing about the output distribution reveals whether such an e was ever known, which is precisely why the provenance of Q matters more than any proof about the generator.

Nothing Up My Sleeve

Dual EC is safe only if nobody knows the relation between P and Q , and the standard never explains where Q came from.

The usual defence is a *nothing-up-my-sleeve* constant: derive the parameter from something with no room for choice, such as the digits of π , and publish the derivation. MD5 uses $\sin$, Blowfish uses π , and the Brainpool curves are generated from hashes of stated seeds.

A plausible-looking constant proves nothing on its own: with enough freedom in *which* innocuous source to use, a designer can search until one of them yields the value they wanted. A seed convinces only if it is short, fixed in advance, and reproducible.

Matthew Green's Blog

- ▶ Dual-EC-DRBG: <https://blog.cryptographyengineering.com/2013/09/18/the-many-flaws-of-dualecdrbg>
- ▶ RSA warning: <https://blog.cryptographyengineering.com/2013/09/20/rsa-warns-developers-against-its-own>
- ▶ NSA random number: <https://blog.cryptographyengineering.com/2013/12/28/a-few-more-notes-on-nsa-random-number>
- ▶ Juniper backdoor: <https://blog.cryptographyengineering.com/2015/12/22/on-juniper-backdoor>



Micali-Schnorr

The same shape, but with RSA instead of a curve: from a state s_i , compute $s_i^e \bmod n$, keep the high bits as the next state, and output the low bits.

The claim is that this inherits the hardness of RSA. It does not, quite: the security rests on a nonstandard assumption, namely that a *truncated* RSA output is indistinguishable from random, which is stronger.

Davis, Green, Heninger, Ryan and Suhl (2023) showed how much room that leaves. For related constructions they exhibit parameters that carry a genuine backdoor, built with multivariate Coppersmith methods, and the generator is standardized in ISO 18031 with parameters nobody has to justify.

No backdoor in the standardized Micali-Schnorr parameters is known...

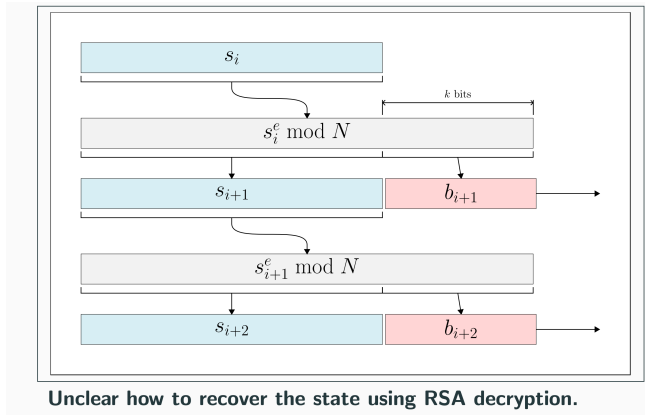


On the Possibility of a Backdoor in the Micali-Schnorr Generator

Hannah Davis¹ Matthew Green² Nadia Heninger¹
Keegan Ryan¹ Adam Suhl¹

Figure: paper: <https://eprint.iacr.org/2023/440.pdf>, talk:
<https://www.youtube.com/watch?v=608NQdTn39Q&t=2629s>, slides:
<https://iacr.org/submit/files/slides/2023/rwc/rwc2023/119/slides.pdf>

Micali-Schnorr?



Contents

Introduction

Symmetric Legacy Crypto

Asymmetric Legacy Crypto

Attacks on TLS < 1.3

Backdoors

Summary

What to Remember

- ▶ Legacy crypto rarely disappears: it survives as a fallback, and a downgrade brings it back
- ▶ Public key schemes fail through parameters and padding, not broken mathematics
- ▶ Symmetric legacy fails in the primitive: collisions, bias, or too small a block
- ▶ Shared parameters concentrate the payoff, as Logjam showed for a few primes
- ▶ TLS 1.3 is a cleanup: fewer choices, all AEAD, and no static key exchange

Questions?