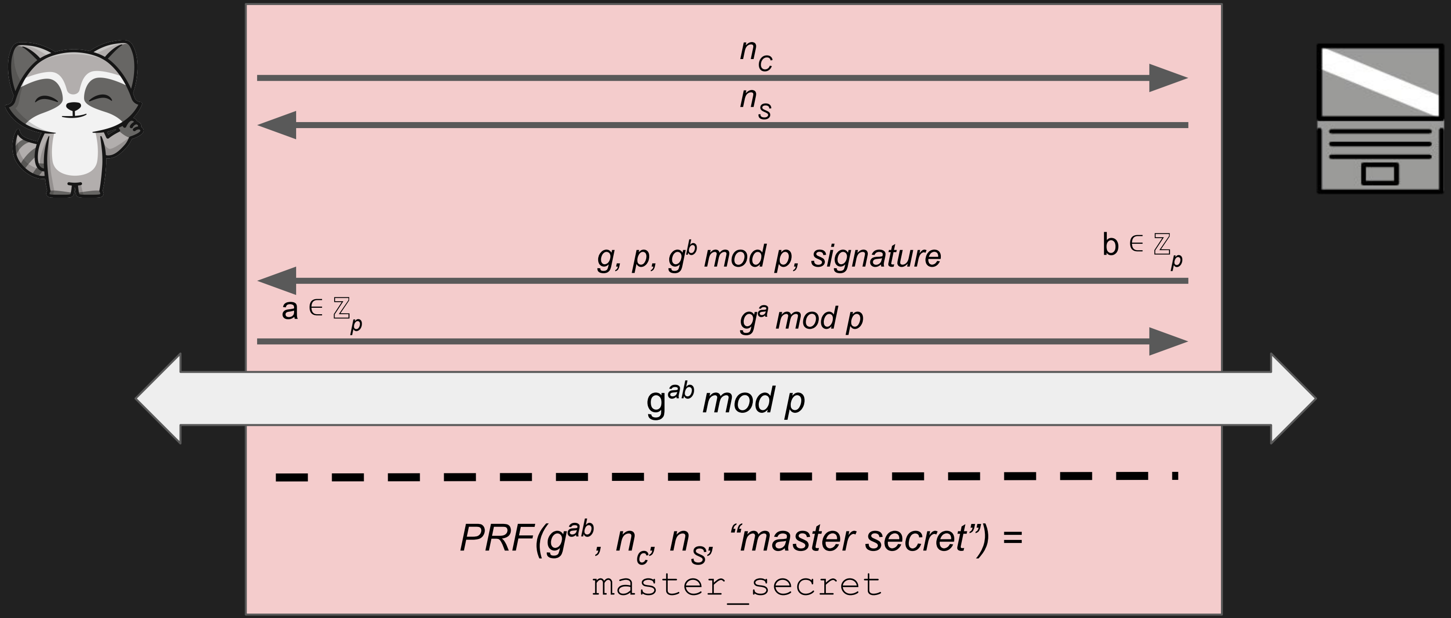
Raccoon Attack: Finding and Exploiting Most-Significant-Bit-Oracles in TLS-DH(E)

Robert Merget,[1] Marcus Brinkmann,[1] Nimrod Aviram,[2]
Juraj Somorovsky,[3] Johannes Mittmann,[4] Jörg Schwenk[1]

Usenix Security 2021

[1] Ruhr University Bochum
[2] School of Computer Science, Tel Aviv University
[3] Paderborn University
[4] Bundesamt für Sicherheit in der Informationstechnik (BSI), Germany

TLS-DH(E)



n_C

n_S

$b \in \mathbb{Z}_p$

g, p, g^b mod p, signature

$a \in \mathbb{Z}_p$

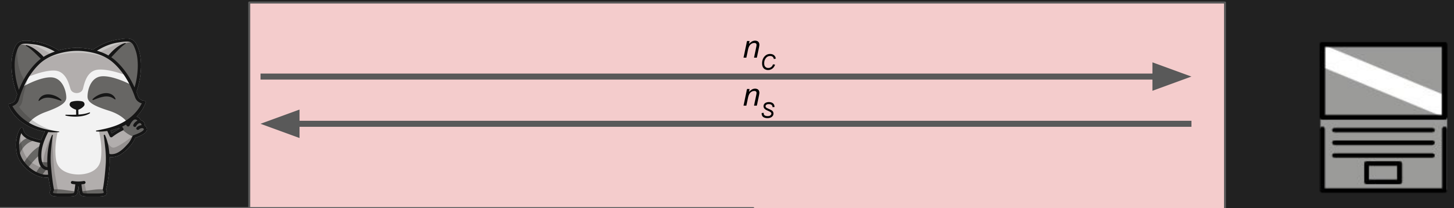
g^a mod p

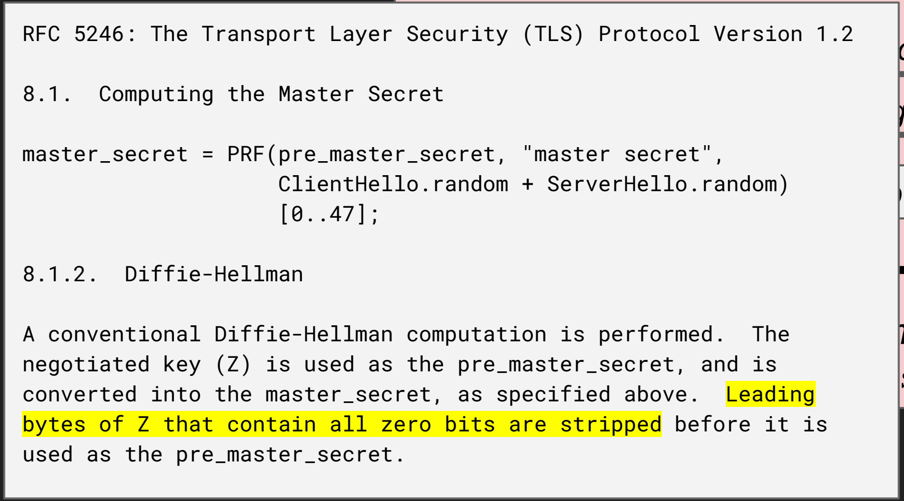
g^{ab} *mod p*

PRF(g^{ab}, n_c, n_S, "master secret") =

`master_secret`

TLS-DH(E)





```
RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2

8.1.  Computing the Master Secret

master_secret = PRF(pre_master_secret, "master secret",
                    ClientHello.random + ServerHello.random)
                    [0..47];

8.1.2.  Diffie-Hellman

A conventional Diffie-Hellman computation is performed.  The
negotiated key (Z) is used as the pre_master_secret, and is
converted into the master_secret, as specified above.  Leading
bytes of Z that contain all zero bits are stripped before it is
used as the pre_master_secret.
```

Some servers reuse ephemeral keys

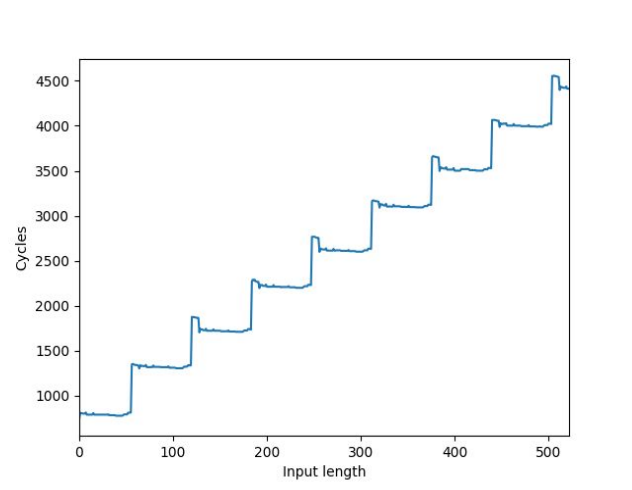
Constant Time Execution

TLS key derivation is based on hash functions

Hash functions operate in O(n) not O(1)

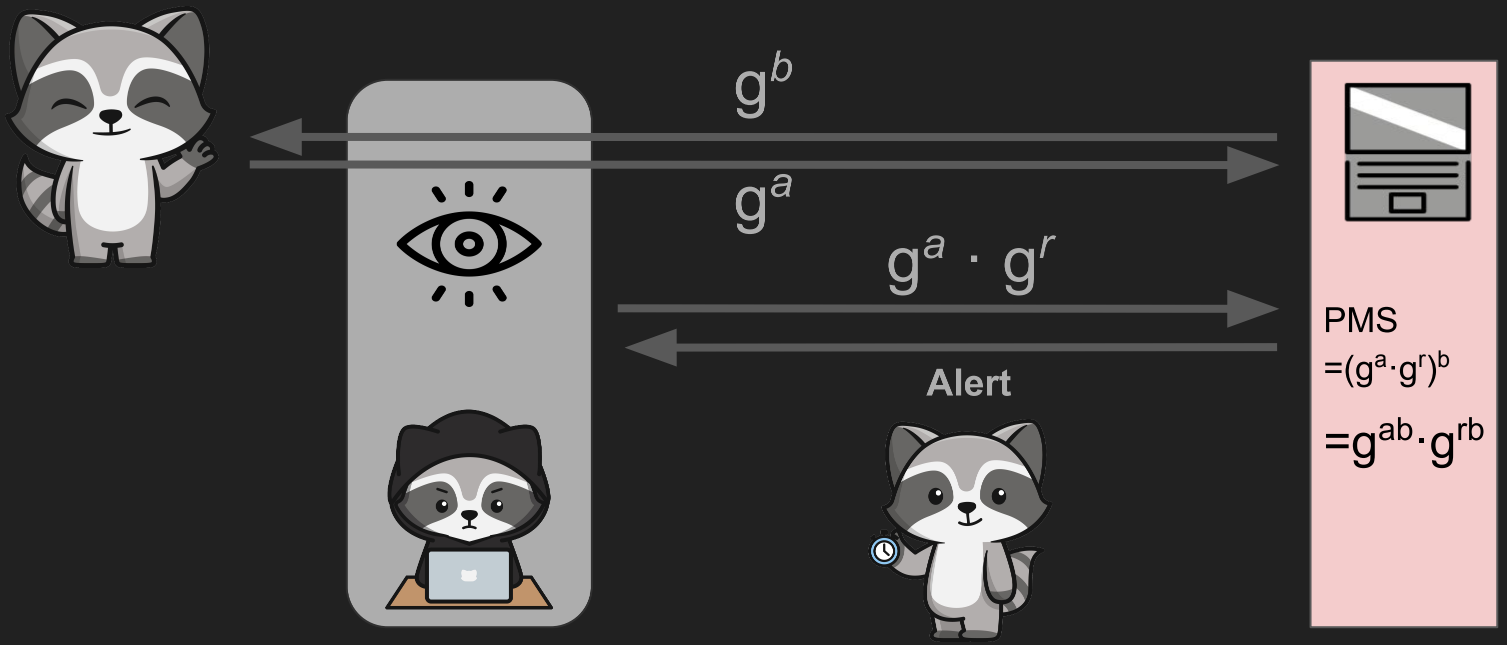
This creates various side-channels:

- Compression function invocation
- Hash function invocation
- Key padding
- Direct side-channel

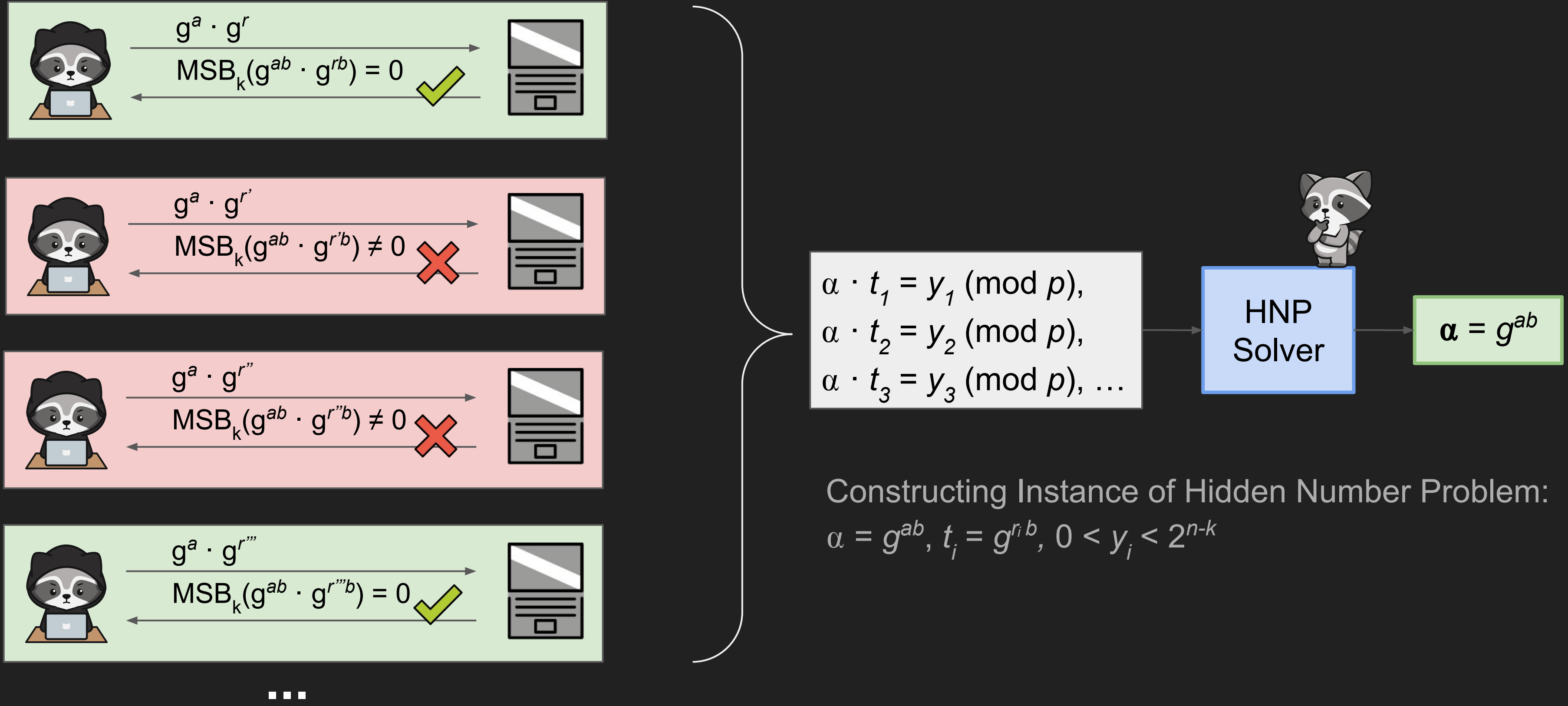


Example: SHA-256 in OpenSSL

Attack Overview



Retrieving the PMS



$g^a \cdot g^r$

$\mathrm{MSB}_k(g^{ab} \cdot g^{rb}) = 0$ ✓

$g^a \cdot g^{r'}$

$\mathrm{MSB}_k(g^{ab} \cdot g^{r'b}) \neq 0$ ✗

$g^a \cdot g^{r''}$

$\mathrm{MSB}_k(g^{ab} \cdot g^{r''b}) \neq 0$ ✗

$g^a \cdot g^{r'''}$

$\mathrm{MSB}_k(g^{ab} \cdot g^{r'''b}) = 0$ ✓

...

$\alpha \cdot t_1 = y_1 \pmod{p}$,
$\alpha \cdot t_2 = y_2 \pmod{p}$,
$\alpha \cdot t_3 = y_3 \pmod{p}$, …

HNP Solver

$\boldsymbol{\alpha} = g^{ab}$

Constructing Instance of Hidden Number Problem:
$\alpha = g^{ab}$, $t_i = g^{r_i b}$, $0 < y_i < 2^{n-k}$

Performance

DH Group	Bit Length	k=24	k=20	k=16	k=12	k=8
RFC 5114	**1024**	d=50 T=6s	d=60 T=10s	d=80 T=26s	d=100 T=111s	d=200 T~=2,5h
LibTomCrypt	**1036**	d=50 T=6s	d=60 T=10s	d=80 T=28s	d=100 T=52s	d=180 T~=1,5h
SKIP	**2048**	d=100 T=112s	d=120 T=207s	d=160 T=977s	Unsolved	Unsolved

k = Leading zero bits leaked
d = Number of equations used
T = Time to solve HNP

Impact

Scan of Alexa Top 100k:

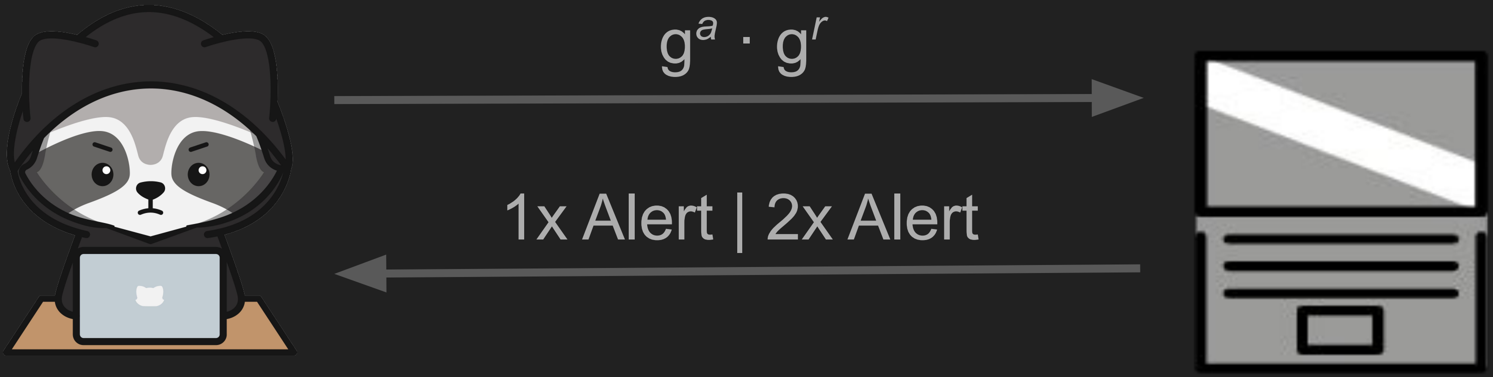
- 32% of the scanned servers supported DHE cipher suites
- 10.9% of those servers reused their ephemeral keys

Firefox was the last browser to drop support in September 2020

No major browser supports DHE anymore

Direct Raccoon

CVE-2020-2529: F5-Big IP leaks leading zero byte



$g^a \cdot g^r$

1x Alert | 2x Alert

Countermeasure

Generally:

- Do not leak partial information about secret values
- Make secrets constant size

For TLS:

- Clients should avoid DH(E)
- Servers should not reuse ephemeral keys
- Servers and clients should not use DH

Raccoon and ECDH(E)

Leading zero bytes of shared ECDH secrets maintained

Requires implementation-specific side-channels

Further research required, currently not exploitable

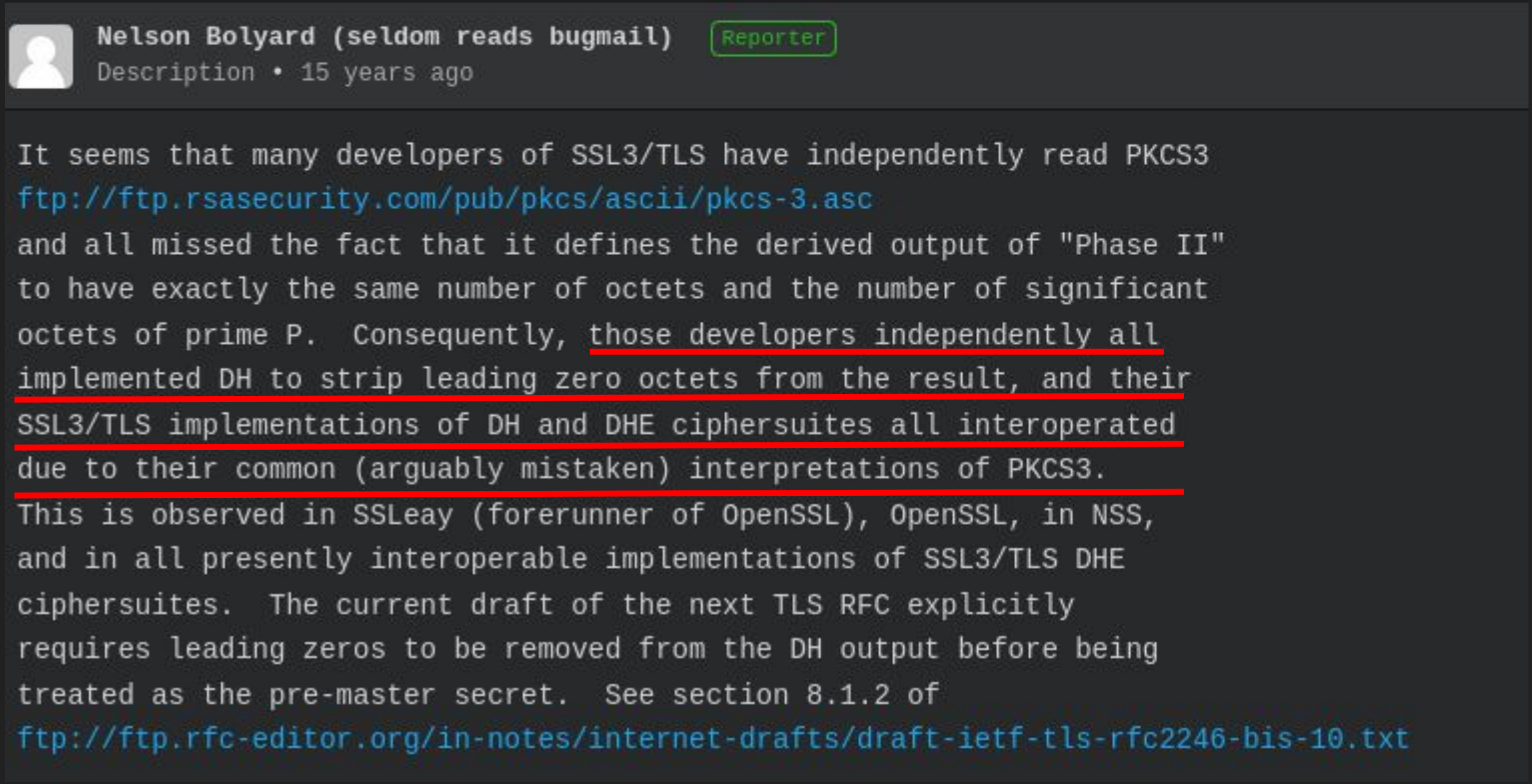
Raccoon and TLS 1.3

Leading zero bytes of ALL shared secrets maintained

Foresight by David Benjamin in Draft-13 proved useful

Ephemeral key reuse is uncommon

Why the mess?



Nelson Bolyard (seldom reads bugmail) Reporter
Description • 15 years ago

It seems that many developers of SSL3/TLS have independently read PKCS3
ftp://ftp.rsasecurity.com/pub/pkcs/ascii/pkcs-3.asc
and all missed the fact that it defines the derived output of "Phase II"
to have exactly the same number of octets and the number of significant
octets of prime P. Consequently, those developers independently all
implemented DH to strip leading zero octets from the result, and their
SSL3/TLS implementations of DH and DHE ciphersuites all interoperated
due to their common (arguably mistaken) interpretations of PKCS3.
This is observed in SSLeay (forerunner of OpenSSL), OpenSSL, in NSS,
and in all presently interoperable implementations of SSL3/TLS DHE
ciphersuites. The current draft of the next TLS RFC explicitly
requires leading zeros to be removed from the DH output before being
treated as the pre-master secret. See section 8.1.2 of
ftp://ftp.rfc-editor.org/in-notes/internet-drafts/draft-ietf-tls-rfc2246-bis-10.txt

Raccoon & DH(E) Proofs

Security proofs exist:

- TLS DH(E) (Jager et. al) (CRYPTO 2012)
- TLS-DH (Krawczyk et. al) (CRYPTO 2013)

Zero byte stripping/timing is not modeled

Proofs rely on PRF-ODH assumption

Assumption is not in practice

Conclusion

- No need to panic, exploitation is difficult
- The Raccoon attack is not TLS specific
- First time HNP is used to attack DH

More info: https://raccoon-attack.com

Tool to scan your servers:

https://github.com/tls-attacker/TLS-Scanner

@ic0nz1
✉ robert.merget@rub.de