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]

Real World Crypto 2021

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

Transport Layer Security

TLS-DH(E)

n_C

n_S

$g, p, g^b \bmod p$, signature

$b \in \mathbb{Z}_p$

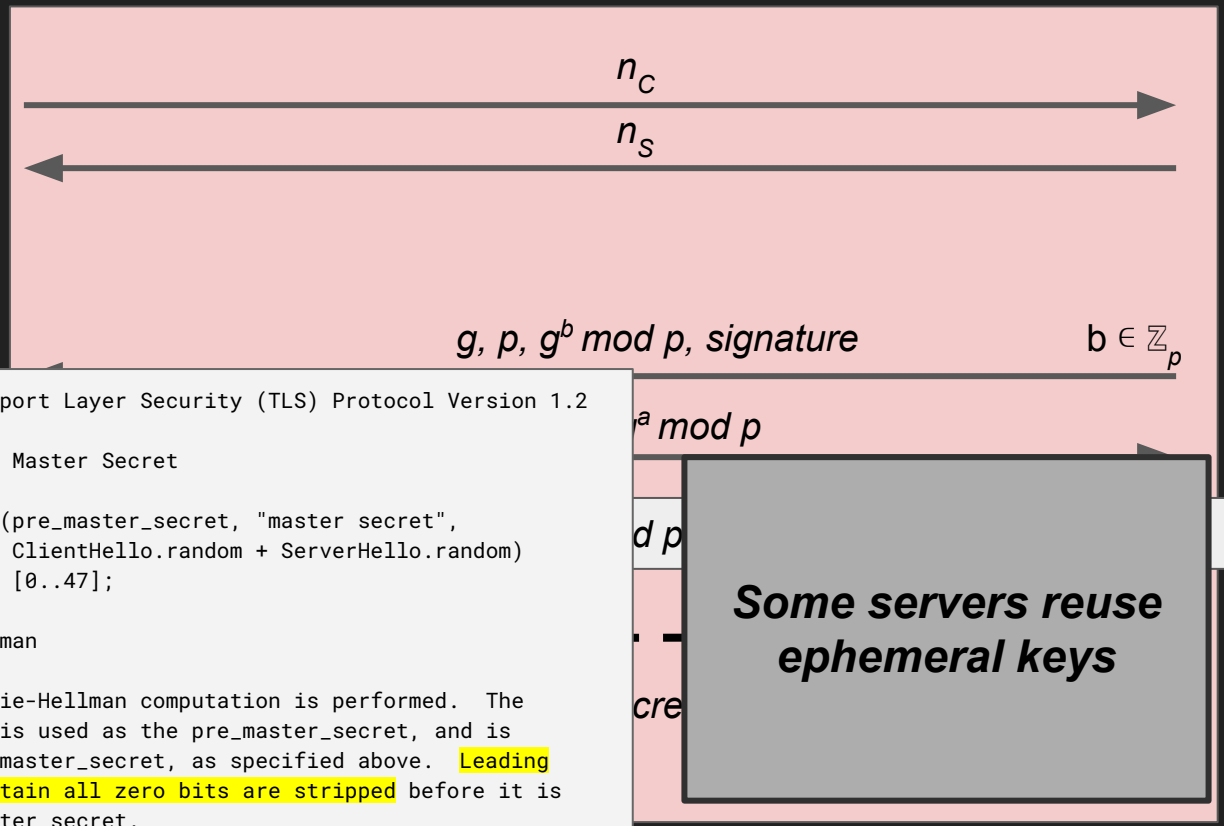
$a \in \mathbb{Z}_p$

$g^a \bmod p$

$g^{ab} \bmod p$

$PRF(g^{ab}, n_c, n_S, \text{“master secret”}) =$ `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.

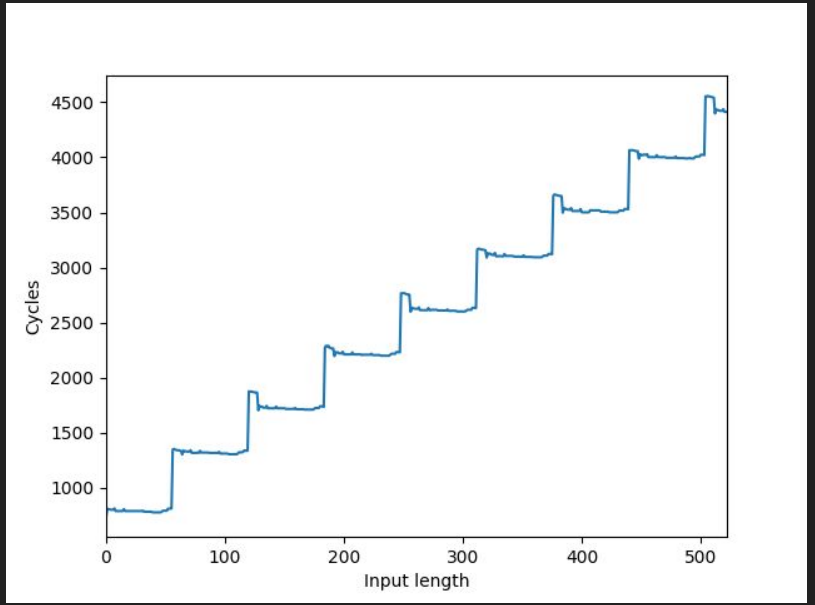
Key derivation and Constant Time Execution

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

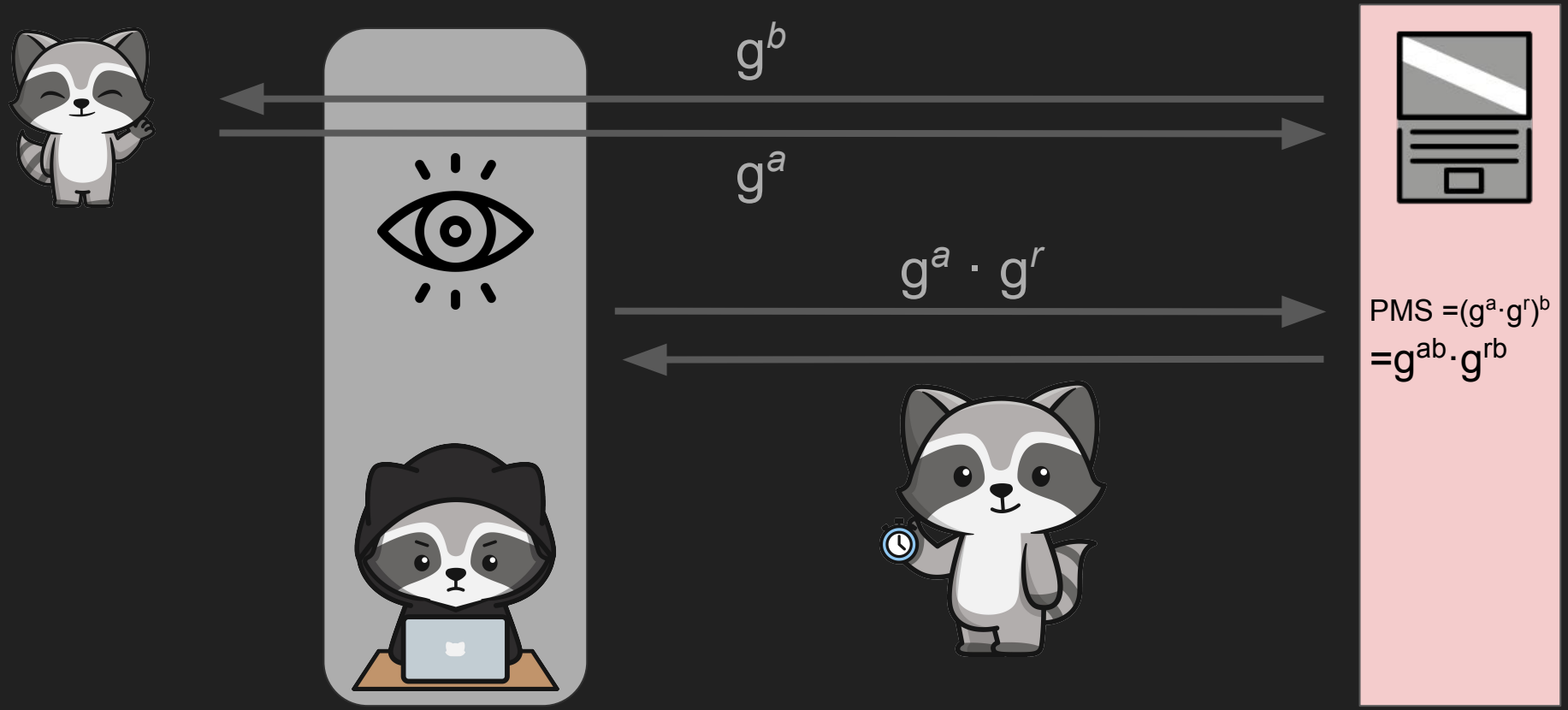
From Side-channel to Exploit

$CKE(g^a)$, *CCS*, FIN_{Inv}

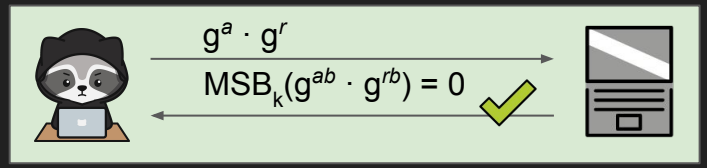
ALERT

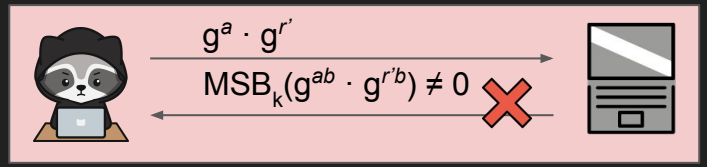
$MSB_k(g^{ab}) = 0?$

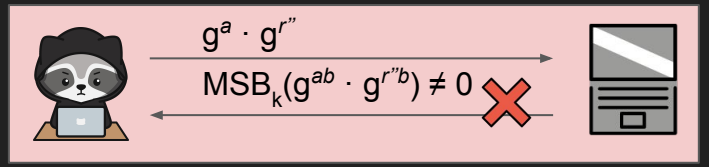
Attack Overview

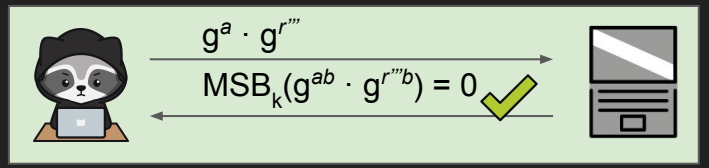


Retrieving the PMS









...

Constructing Instance of Hidden Number Problem:

$\alpha = g^{ab}$, $t_i = g^{r_i b}$, $0 < y_i < 2^{n-k}$

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

HNP Solver

$\alpha = g^{ab}$

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=9295s
LibTomCrypt	**1036**	d=50 T=6s	d=60 T=10s	d=80 T=28s	d=100 T=52s	d=180 T=5613s
SKIP	**2048**	d=100 T=112s	d=120 T=207s	d=160 T=977s	Unsolved	Unsolved

d = Number of equations required
T = Time required to solve HNP
k = Leading Zero bits leaked

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 to drop support in September 2020

No major browser supports DHE anymore

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)

TLS does not strip leading zero bytes of shared ECDH secrets

Requires implementation specific side-channels

Further research required, currently not exploitable

Raccoon and TLS 1.3

TLS 1.3 does not strip leading zero bytes of shared secrets

Foresight by David Benjamin in Draft-13 proved useful

Ephemeral key reuse is uncommon

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