TLS

THE LAW OF BROKEN TLS IMPLEMENTATIONS

Hanno Böck
https://hboeck.de/

WHO AM I?

Hanno Böck

Freelance journalist (Golem.de, Zeit Online, taz, LWN)

Find and fix security vulnerabilities and bugs in free software (Fuzzing Project, supported by Linux Foundation's Core Infrastructure Initiative)

Monthly Bulletproof TLS Newsletter

EVERY IMAGINABLE TLS IMPLEMENTATION FLAW CAN BE FOUND IN THE WILD.

TLS IMPLEMENTATION BUGS

OMMITTING CHECKS

TLS implementations should check various things to assure correctness of connection, like:

- Padding (since TLSv1.0)
- MAC / authentication tag
- FinishedMessage

POODLE TLS

The POODLE attack relies on the undefined padding of SSLv3 - TLSv1.0 defines padding.

Some implementations didn't check padding.

Affects F5, A10, Fortinet, Cisco, IBM, Juniper

"The POODLE bites again", Adam Langley (2014)

MORE POODLES

Maybe you check only some bytes of the padding?

Cisco (Cavium), Citrix, GnuTLS

"There are more POODLEs in the forest", Yngve Petterssen (2015)

MAC / FINISHEDMESSAGE

MACE: Completely ommit MAC check (no authentication).

F5, Cisco, Fortinet

Don't check FinishedMessage (protects handshake).

F5, Juniper

"The POODLE has friends" (Pettersen, 2015)

GCM NONCE REUSE

GCM needs a nonce value.

If one uses the same nonce and key twice everything falls appart.

TLS / GCM NONCES

TLS gives no guidance how to select a nonce. A counter is secure.

Some implementations get it wrong: Duplicate nonces (Radware, Cavium), random nonces (IBM, A10, Sangfor).

"Nonce-Disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS", Böck, Zauner, Devlin, Somorovsky, Jovanovic (2016)

DIFFIE HELLMAN PARAMETERS

Diffie Hellman uses a set of parameters - a prime and a generator.

But sometimes the prime isn't prime - why?

DIFFIE HELLMAN NON-PRIME

socat: new parameters, "prime" is not prime (CVE-2016-2217).

Could that be a backdoor? (see Wong, 2016)

Internet-wide scan found 500 IPs with non-prime (Dorey, Chang-Fong, Essex 2016)

DIFFIE HELLMAN PARAMETER CONFUSION

However, some servers in our scans used Java's DSA primes as p but mistakenly used the DSA group order q in the place of the generator g. We found 5,741 hosts misconfigured this way.

This substitution of q for g is likely due to a usability problem: the canonical ASN.1 representation of Diffie-Hellman key exchange parameters (coming from PKCS#3) is a sequence (p, g), while that of DSA parameters (coming from PKIX) is (p, q, g); we conjecture that the confusion between these formats led to a simple programming error. (Logjam paper, 2015)

SAFE PRIMES

One can use "safe primes" for DH.

Or other primes, but then one can't reuse the ephemeral key (OpenSSL, CVE-2016-0701).

DH PARAMETER CHECKING IMPOSSIBLE

Diffie Hellman parameters can contain undetectable backdoors that cannot be prevented with parameter checking (Fried, Gaudry, Heninger, Thome 2016).

RSA-CRT

CRT-optimization of RSA: Split private key signature operation into two calculations.

Dangerous: If one calculation produces wrong result this leaks the private key.

Citrix, Hillstone Networks, ZyXEL, Radware (all Cavium chip), Alteon/Nortel, Viprinet, QNO, BEJY

"Factoring RSA Keys With TLS Perfect Forward Secrecy", Florian Weimer (2015)

MATH

Cryptography is based on mathematics.

Math libraries can have bugs.

CVE-2014-3570

Bug in BN_sqr() function of OpenSSL.

Produces wrong results in some cases.

DIFFERENTIAL FUZZ-TESTING

CVE-2015-3193: bug in BN_mod_exp() / OpenSSL

CVE-2016-1938: bug in mp_div()/mp_exptmod() / NSS

CVE-2015-8803/CVE-2015-8804: Bugs in elliptic curve multiplications / Nettle

Recently: Several bugs in Poly1305 / OpenSSL

CVE-2016-6885/CVE-2016-6886/CVE-2016-6887/CVE-2016-8671: pstm_exptmod() / MatrixSSL

Usually carry propagation bugs, all thanks to american fuzzy lop.

BUGS IN CALCULATIONS

Hard or impossible to test remotely.

Even testing without source tricky.

HANDSHAKE SIZE

With new extensions and ciphers the TLS handshake grew.

F5 load balancers couldn't handle handshakes larger than 256 bytes.

"If you use F5/BIG-IP devices to terminate SSL connections, please update the firmware on the things! We're trying to run an Internet here and old versions of these devices are a real problem for deploying new TLS features." (Adam Langley, 2013)

F5 HANDSHAKE PROBLEM

It turned out F5 load balancers fail with handshakes between 256 and 512 bytes.

Solution: If your handshake is bigger than 256 bytes pad it to be bigger than 512 bytes.

TLS Padding Extension (RFC 7685).

AND THEN...

There are other TLS implementations that fail with handshakes bigger than 512 bytes (Cisco Ironport).

The following bug workaround options are available:

SSL_OP_SSLREF2_REUSE_CERT_TYPE_BUG

…

SSL_OP_MICROSOFT_BIG_SSLV3_BUFFER

…

SSL_OP_SAFARI_ECDHE_ECDSA_BUG

Don't prefer ECDHE-ECDSA ciphers when the client appears to be Safari on OS X. OS X 10.8..10.8.3 has broken support for ECDHE-ECDSA ciphers.

SSL_OP_SSLEAY_080_CLIENT_DH_BUG

…

SSL_OP_TLS_D5_BUG

…

SSL_OP_DONT_INSERT_EMPTY_FRAGMENTS

Disables a countermeasure against a SSL 3.0/TLS 1.0 protocol vulnerability affecting CBC ciphers, which cannot be handled by some broken SSL implementations. This option has no effect for connections using other ciphers.

SSL_OP_TLSEXT_PADDING

Adds a padding extension to ensure the ClientHello size is never between 256 and 511 bytes in length. This is needed as a workaround for some implementations.

DOWNGRADES

Hanno Boeck 2008-08-12 09:55:58 PDT Description

```
User-Agent:       Mozilla/5.0 (X11; U; Linux i686; de; rv:1.9.0.1) Gecko/2008072610 Firefox/3.0.1
Build Identifier: Mozilla/5.0 (X11; U; Linux i686; de; rv:1.9.0.1) Gecko/2008072610 Firefox/3.0.1

When using Firefox on a slow / unreliable internet connection (e.g. GPRS), it often fails to process
SNI correctly and the user is left with the main certificate which would be delivered without SNI.

This should be considered serious as it'll leave the user with a certificate issued for the wrong
domain and will give him a warning looking like someone was trying to attack him.

Reproducible: Sometimes
```

SSL/TLS VERSIONS DURING HANDSHAKE

ClientHello: "Dear server, the maximum version I support is TLSv1.2"

ServerHello: "I don't support that new TLSv1.2 stuff, let's use TLSv1.0"

BUT SOMETIMES...

ClientHello: "Dear server, the maximum version I support is TLSv1.2"

Server thinks: "I never heard of TLSv1.2... Maybe I better say nothing at all or send an error..."

Version intolerance (this is always a server bug)

IT'S AN OLD ISSUE

Known at least since 2003.

WHAT BROWSERS DID

Browser tries to connect with TLSv1.2.

No answer? Browser retries with TLsv1.1, TLSv1.0, SSLv3.

Retries all supported versions.

Behavior has been called "Protocol Dance".

THE SNI BUG

Sometimes bad internet connections caused downgrade from TLSv1.0 to SSLv3 (1.1/1.2 wasn't implemented yet).

SSLv3 does not support SNI - therefore wrong certificate.

My server was behaving fine - but Mozilla refused to fix it, because they wanted to retain compatibility with broken servers.

BLACK HAT 2014

Antoine Delignat-Lavaud presents Virtual Host Confusion attack.

Prevent SSL Downgrading. Current browsers attempt to maximize their compatibility with buggy TLS implementations by retrying failed handshakes with downgraded TLS versions, all the way from TLS 1.2 to SSL3. There has been concerns about this behavior, notably because the newest cipher suites are only available in TLS 1.2.

It turns out that it can also be taken advantage of by a network attacker to exploit virtual host confusion attacks.

POODLE (2014)

Padding Oracle On Downgraded Legacy Encryption

Another Padding Oracle that only works against SSLv3.

Good for the attacker: We can downgrade users.

SOLUTION

SCSV (RFC 7507): Server signals browser that it is not broken.

PROTOCOL DANCE AND SCSV

1. We have a version negotiation mechanism
2. Servers have broken TLS implementations.
3. Browsers implement workaround.
4. Workaround introduces security issue.
5. Workaround for security issue introduced by workaround gets standardized.

VERSION INTOLERANCE

Issue was known and documented since at least 2003.

By now most browser downgrades have been removed.

But what about TLS 1.3?

VERSION INTOLERANCE

"It's taken about 15 years to get to the point where web browsers don't have to work around broken version negotiation in TLS and that's mostly because we only have three active versions of TLS. When we try to add a fourth (TLS 1.3) in the next year, we'll have to add back the workaround, no doubt." Adam Langley (2016)

VENDORS

IBM: "I expect both releases towards the end of the year as 8.5.5.10 and 9.0.0.1 are already at the tail end of their release processes.

Citrix: "Our investigation indicates that this is not a security issue. We also have this issue on our radar and plan to address it in an upcoming Citrix NetScaler version."

Cisco: "when it comes to devices or releases that have passed the last day of support (not the end of life), we can't do anything about them."

SITE OPERATORS

apple.com: no reply

paypal.com: "SSL issues are out of scope for PayPal Bug Bounty Program"

ebay.com: no reply

TLS 1.3 NEW VERSION NEGOTIATION

Old version field gets deprecated.

List of versions in an extension.

Caveat: Instead of two version numbers from which one is useless we'd then have three version numbers from which two are useless.

GREASE

GREASE (Generate Random Extensions And Sustain Extensibility), proposal by David Benjamin (Google).

Reserve garbage values for version numbers (and ciphers, extensions, ...) that get sent ocassionally to make sure implementations don't mess things up too badly.

GREASE PARADIGM

Design new protocols in a way that they can get deployed despite a broken ecosystem.

Create protocols that are hard to mess up.

WILL GREASE WORK?

It's still possible to create version intolerance in a GREASE-scenario: Just whitelist the GREASE values.

Will there be vendors that are so stupid? We'll see.

THANKS FOR LISTENING

Questions?

https://hboeck.de/