Some tales about TLS

Hanno Böck
https://hboeck.de/

CC 0 ZERO

Last year

- Last year: "How broken is TLS?"
- The good news: Things are definitely improving

CC 0 ZERO

Why care?

- TLS is *the* most important cryptographic protocol in the Internet
- TLS is under attack: BEAST, CRIME, Lucky Thirteen, Heartbleed, BERserk, POODLE, FREAK

CC 0 ZERO

CA issues all the time

- June 2013: ANSSI issues certs for Google
- March 2014: India CCA intermediate compromised and issued certs for Yahoo and Google
- Feb 2015: Superfish / Privdog / Komodia breaking certificate authentication
- March 2015: Comodo cert for live.fi through hostmaster@live.fi
- March 2015: Same thing for xs4all
- March 2015: Google found bad certs issued by MCS Holdings / CNNICa
- April 2015: Google and Mozilla remove CNNIC

CC 0 ZERO

Too many CAs

- There are hundreds of browser-accepted CAs and an unknown number of subordinate CAs
- Each of them can break TLS security
- It does not matter how good your CA is - the only thing that matters is the worst CA from all of them

CC 0 ZERO

CNNIC

- CNNIC issued intermediate certificate to egyptian company MCS Holdings
- MCS used it in a Man-in-the-Middle-TLS-Proxy in violation of policy
- Google and Mozilla kick CNNIC out

CC 0 ZERO

Domain Validation via E-Mail

- Domain Validation: CA sends mail to defined aliases (admin, administrator, webmaster, hostmaster, postmaster, see Baseline Requirements)
- If you offer E-Mail you **must** make sure that noone can register such an address
- One can argue if this is a sane system, but it's clearly documented (Baseline Requirements)
- live.fi / xs4all.nl issues were their fault

CC 0 ZERO

Revocation is broken

- Two revocation mechanisms: CRL (impractical) and OCSP
- Browsers used insecure soft-fail mode in the past
- Chrome and Firefox distribute their own blocklists, but they don't scale
- OCSP stapling could help, but needs a mechanism to indicate its use (muststaple draft)

CC 0 ZERO

Man in the Middle Proxies

- Superfish: Created a TLS Man in the Middle Proxy, private key was static and part of the Software (Komodia)
- Privdog: Just disabled TLS verification completely (Privdog is founded by the CEO of Comodo)
- Several Antiviruses do the same. Not fully broken, but all decrease the security of TLS
- This is not directly a problem of CAs or TLS
- TLS Man in the Middle Proxies are a bad idea

CC 0 ZERO

Alternatives and Mitigations

- It is easy to point out the flaws of the CA system, much harder to offer an alternative
- Complaining and using no encryption at all doesn't help
- With all its downsides, there is one pretty strong argument for the CA system: It's usable

CC 0 ZERO

DNSSEC/DANE

DANE won't provide you any security today

It is very uncertain if it will ever do that

CC 0 ZERO

DNSSEC - too many pieces

- For DNSSEC to work you need
  - A signed root
  - A signed Top Level Domain
  - A domain broker that supports DNSSEC
  - A DNS operator that supports DNSSEC
  - A client that verifies DNSSEC
- Only if you have all 5 you have security

CC 0 ZERO

Working DNSSEC deployment is near zero

- DNSSEC propaganda: "xx % of all TLDs are signed", "there are already XX.XXX signed domains"
- Completely irrelevant statements
- Cryptographic signatures aren't worth anything if nobody is checking them
- Client deployment of DNSSEC is very close to zero

CC 0 ZERO

DNSSEC client

- So how exactly does a client verify DNSSEC signatures? (Most common today: Not at all)
- DNSSEC verification happens in the DNS resolver - but clients usually don't have DNS resovlers

CC 0 ZERO

DNSSEC client

- Should we trust our providers? (No!)
- Should operating systems ship DNS resolvers?
- Should applications ship their own DNS resolvers?
- It's not even clear how DNSSEC should be deployed on clients

CC 0 ZERO

More DNSSEC problems

- Reflection / amplification attacks (fixable, but people don't fix it)
- Bad crypto (fixable, but people don't fix it)
- Still hiearchical, moving trust to TLD operators (essentially that means nation states)
- Almost impossible to revoke trust

CC 0 ZERO

So what is DANE

- Idea of DANE: If we already have a secure DNS through DNSSEC we can add certificate information to the DNS
- The problem: We don't have working DNSSEC
- Building something on top of something that does not work is pointless

CC 0 ZERO

DNSSEC - other voices

- "DNSSEC is undeployable in practice. It is a state far worse than IPv6, and no amount of wishing or application activism is going to change this - it's a problem of economy, not education." Ryan Sleevi, Chrome-developer, Google
- "DNSSEC is dead", "DNSSEC is not maintainable at scale and not end-to-end.", "I looked into what we would have to do to run DNSSEC on our millions of domains. Not fun, no benefit, we become DDoS source." Alex Stamos, Yahoo CSIO
- "If you're running systems carefully today, no security problem you have gets solved by deploying DNSSEC." Thomas Ptacek, crypto expert

CC 0 ZERO

HTTP Public Key Pinning (HPKP)

- Webpage sends a header with hashes of public keys for the browser to pin
- Browser stores these hashes
- Always needs at least two keys - because you need to be able to change your certificates in the future
- Adds a "Trust on First use" (ToFU) protection

CC 0 ZERO

HTTP Public Key Pinning (HPKP)

- HPKP header:
- max-age=31536000;pin-sha256="HD3EpAqgxJWKGiSuuXPyipmL33IwYlwhLUgF1gKYOuc=" sha256="dwUkkREEnv6pEtNJoRzlBHJm3IlUvPhgy0mdYFOM6V8=" includeSubDomains; report-uri="/hpkp.php"
- Browser pins the two hashes for [max-age] seconds
- report-uri is unimplemented today

HPKP deployment

- HPKP is supported by Chrome/Chromium and Firefox
- Needed for deployment: Software change in browsers and configuration change on servers (compare that to DANE)
- Large webpages have pre-loaded pins in the browsers

CC 0 ZERO

HPKP: Only for HTTPS

- One big drawback: It is only for the web
- As HPKP is implemented via HTTP headers it does not work on other protocols
- There was a proposal called TACK to do something very similar on the TLS layer, but its development is stale at the moment

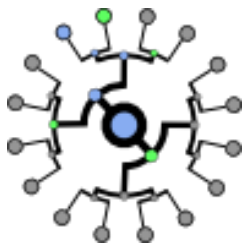
CC 0 ZERO

HPKP Warning

- HPKP adds a lot of security, but it can be dangerous
- If you loose your keys you may lock out your visitors
- Needs careful planning of key management

CC 0 ZERO

Certificate Transparency



- Public logs with all certs in them
- Certificate can contain log proof confirming that it has been added to a log
- When a browser sees a certificate that is not in the log it can raise alarm

CC 0 ZERO

Certificate Transparency

- Certificate Transparency will run in soft-fail mode, it can't prevent misuse
- But it makes it very hard to use malicious certificates without being noticed
- Plan from Google: Require CT for EV certs, later for all certs

CC 0 ZERO

Free certificates

- StartSSL, free for noncommercial use, 1 year validity (received criticism because they charged for revocation after Heartbleed)
- WoSign - since February 2015, 2 years validity
- Let's encrypt - will start in summer (EFF, Mozilla, cross-signed by IdenTrust)

CC 0 ZERO

Ciphers

- "This seems like a good moment to reiterate that everything less than TLS 1.2 with an AEAD cipher suite is cryptographically broken." Adam Langley, Google
- Only TLS 1.2 with GCM cipher suites is really safe

CC 0 ZERO

Cipher suites

- A good cipher suite: ECDHE-RSA-AES128-GCM-SHA256
- A really bad cipher suite: EXP-RC2-CBC-MD5
- Three things: Public key algorithm, key exchange and symmetric cipher

CC 0 ZERO

Public key algorithms

- There are RSA, DSA and ECDSA
- RSA is the default
- DSA is not used at all
- ECDSA is used by some big players (Google, Cloudflare), not trivial to get certificate

CC 0 ZERO

Key exchange

- Classic RSA exchange: Should not be used any more, does not provide Forward Secrecy
- Diffie Hellman and Elliptic Curve Diffie Hellman
- Diffie Hellman with 1024 bit is weak, but widely in use (apache before 2.4.7 doesn't support larger DH exchange)
- Elliptic curves: Some (very vague) doubts about NIST curves

CC 0 ZERO

Block ciphers with CBC

- Up to TLS 1.1 all block ciphers in TLS used CBC with MAC-then-Encrypt
- Up to TLS 1.0 with an imlicit IV - this lead to the BEAST attack
- If attacker can separate padding errors from MAC errors this leads to vulnerabilities (Padding Oracle, Lucky Thirteen)
- Encrypt-then-MAC-extension, rarely used (RFC 7366)

CC 0 ZERO

They knew there was a problem

> *This leaves a small timing channel, since MAC performance depends to some extent on the size of the data fragment, but it is not believed to be large enough to be exploitable, due to the large block size of existing MACs and the small size of the timing signal.*

TLS 1.2, RFC 5246

CC 0 ZERO

RC4

- RC4 is now officially declared dead (RFC 7465)
- RC4 keystream is biased on certain bits (Mantin, Shamir 2001)
- Practical attack on TLS 2013 (Patterson, Bernstein et al)
- Some new attacks in 2015, especially IMAP and HTTP Basic Auth vulnerable

CC 0 ZERO

AES-GCM

- With RC4 and CBC weak there's only TLS 1.2 with AES-GCM modes left
- Not practical to require GCM modes
- CBC mode problems can be mitigated, so it needs to stay on
- Shipping hard- or software today that does not support TLS 1.2 / AES-GCM should be considered malpractice (Apple Safari!)

CC 0 ZERO

POODLE

- SSLv3 has non-strict padding, which allowed a variant of the padding oracle attack
- Also TLS clients that don't check padding were found (F5, Cisco, Juniper, IBM, Erlang, ...)
- SSLv3 is from 1996 - why is this a problem?
- In theory server and client should negotiate the best protocol both support

CC 0 ZERO

Protocol dance

- Browsers implemented fallbacks that are now called "Protocol Dance"
- If server doesn't answer let's retry with all older protocols (TLS1.2 - TLS1.1 - TLS1.0 - SSLv3)
- A bad workaround that causes security problems (not the first one - Virtual Host Confusion)
- Now there is a workaround for the workaround: SCSV, where the server can indicate that it is not broken
- Mozilla removed protocol dance after POODLE - thank you!

CC 0 ZERO

Disabling SSLv3

- So we just disable SSLv3 and be done with it? Well...
- Microsoft/Nokia shipped high end phones in 2010 with a mail client not supporting anything better than SSLv3
- Similar problem with AVM FritzBox
- How can we stop companies from doing this again? TLS 1.0 may become problematic

CC 0 ZERO

BERserk

- BERserk was a catastrophic failure in the certificate validation of the NSS library (used by Firefox / Chrome)
- Never got the attention it deserved, because it was published the same day as Shellshock

CC 0 ZERO

Bleichenbacher signature forgery attack comes back

- 2006: Bleichenbacher signature forgery attack (not to be confused with Bleichenbacher 98 attack on RSA encryption)
- RSA PKCS #1 1.5 looks like this:
  `00 01 FF FF ... FF 00 ASN.1 HASH`
- Original Bleichenbacher attack: If something comes behind hash we can forge the signature
- BERserk 2014: Ambiguous encoding of ASN.1 identifier allows similar attack
- Only works with PKCS #1 1.5 (2.1 was released 2002) and with very small exponents (typically e=3)

CC 0 ZERO

Other attacks

- CRIME, BREACH: Attacking compression
- CCS injection: State machine issue
- Tripe Handshake: issues regarding handshakes and renegotiation (mainly affects client certificates)
- Virtual Host Confusion: Is TLS host and HTTP host the same?
- Cookie Clutter: truncation issues
- SMACK / FREAK: State machine issues

CC 0 ZERO

Other browsers

- Chrome is leading and Firefox is following
- Internet Explorer and Safari not so much...
- Internet Explorer supports no HSTS, no HPKP, no downgrade protection
- Safari supports no GCM (the only secure cipher these days), no HPKP, no downgrade protection
- Most "alternative" browsers on Linux (Epiphany, Konqueror, Rekonq, ...) have essentially no real security support - the only reason they're not owned all the time is they're rarely used

CC 0 ZERO

Implementations

- It became fashionable to rant about OpenSSL - people tend to ignore that all major TLS implementations had severe bugs in 2014
- OpenSSL is getting much better - the bugs found these days are often really obscure
- Some people try to reimplement TLS in safer languages (miTLS, ocaml-tls) - we'll see how that plays out

CC 0 ZERO

HTTPS by default

- Many people lately are pushing for HTTPS by default
- Cloudflare provides HTTPS for all their pages (using SNI and ECDSA certificates)
- Google wants to mark HTTP pages insecure and raises search engine rank

CC 0 ZERO

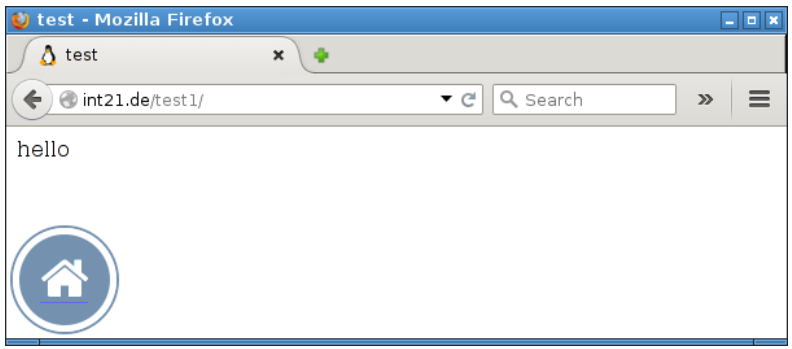
Performance

- Performance problems of TLS are largely based on urban legends, not on facts
- *In January this year (2010), Gmail switched to using HTTPS for everything by default. [...] In order to do this we had to deploy no additional machines and no special hardware. On our production frontend machines, SSL/TLS accounts for less than 1*

  Adam Langley, Google

CC 0 ZERO

Not needed?



- This is a dummy html page at Brussels train station
- HTTP interception widespread: ad injection, tracking cookies
- HTTPS doesn't only encrypt, it also provides content integrity

CC 0 ZERO

Only for logins?

- Some pages only encrypt the login, not the page itself (ebay, amazon)
- This is fully open to SSL Stripping Attacks and has a range of other issues
- There is no way to make such a setup secure

CC 0 ZERO

Why reject HTTPS?

- Poor understanding of TLS, urban legends
- But also: External content
- Ad networks biggest showstopper for HTTPS deployment these days
- This is why news webpages rarely do HTTPS

CC 0 ZERO

HTTP Strict Transport Security (HSTS)

- HSTS tells the browser to mark a page as HTTPS only for a defined timeframe
- Further prevents stripping attacks
- You can even pre-load your webpage as HTTPS only into Chrome and Firefox

CC 0 ZERO

HSTS attack through NTP

- HSTS protects a page for a defined timeframe
- System time is considered trustworthy, but it isn't!
- Delorean-Attack - circumvent HSTS with NTP (Selvi 2014)
- NTP provides no security (solutions: tlsdate, openntpd)

TLS 1.3

- TLS working group is busy creating version 1.3 - to be expected late 2015
- Many things still not decided
- Improvements: Less round trips to reduce latency, no RSA key exchange, only authenticated encryption ciphers
- After a long battle: Curve25519

CC 0 ZERO

Quantum computers

- Quantum computers endanger all public key and key exchange algorithms in use today
- Post-quantum Cryptography:Hash-based algorithms (SPHINCS) and lattice-based algorithms (Ring Learning With Errors) are promising, but more research is needed
- Many post-quantum algorithms have large keys, large signatures or other disadvantages

CC 0 ZERO

Use HTTPS

- Use HTTPS - on every webpage
- Require TLS for POP3, IMAP, Jabber, ...
- Support TLS 1.2 and GCM
- Disable SSLv2/3, RC4, TLS Compression
- Use HSTS, OCSP stapling, HPKP and soon Certificate Transparency
- Use the Qualys SSL Labs Test

CC 0 ZERO

Sources I

- How broken is TLS? http://media.ccc.de/browse/conferences/eh2014/EH2014_-_5744_-_de_-_shack-seminarraum_-_20140420153O_-_wie_kaputt_ist_tls_-_hanno.html
- Google on CNNIC http://googleonlinesecurity.blogspot.com/2015/03/maintaining-digital-certificate-security.html
- Mozilla on CNNIC https://blog.mozilla.org/security/2015/04/02/distrusting-new-cnnic-certificates/
- live.fi bad cert https://technet.microsoft.com/en-us/library/security/3046310

Sources II

- xs4all bad cert https://raymii.org/s/blog/How_I_got_a_valid_SSL_certificate_for_my_ISPs_main_website.html
- OCSP muststaple https://tools.ietf.org/html/draft-hallambaker-muststaple-00
- Superfish https://noncombatant.org/2015/02/21/superfish-round-up/
- Privdog https://blog.hboeck.de/archives/865-Software-Privdog-worse-than-Superfish.html
- Why not DNS records (Ryan Sleevi) https://lists.w3.org/Archives/Public/public-webappsec/2014Dec/0264.html

CC 0 ZERO

Sources III

- DNSSEC is dead (Alex Stamos) http://www.slideshare.net/astamos/appsec-is-eating-security
- Against DNSSEC (Thomas Ptacek) http://sockpuppet.org/blog/2015/01/15/against-dnssec/
- HPKP https://developer.mozilla.org/en-US/docs/Web/Security/Public_Key_Pinning
- HPKP draft https://tools.ietf.org/html/draft-ietf-websec-key-pinning-21
- HPKP script for spki hashes https://github.com/hannob/hpkp
- Certificate Transparency http://www.certificate-transparency.org/

CC 0 ZERO

Sources IV

- StartSSL `https://www.startssl.com/`
- Wosign `https://wosign.com/`
- Let's encrypt `https://letsencrypt.org/`
- POODLE bites again `https://www.imperialviolet.org/2014/12/08/poodleagain.html`
- TLS 1.2 / RFC 5246 `https://www.ietf.org/rfc/rfc5246.txt`
- Encrypt-then-MAC / RFC 7366 `https://tools.ietf.org/html/rfc7366`
- RC4 attacks 2013 `http://www.isg.rhul.ac.uk/tls/`

CC 0 ZERO

Sources V

- RC4 attacks 2015 IMAP / HTTP Basic Auth `http://www.isg.rhul.ac.uk/tls/RC4mustdie.html`
- RC4 Bar Mitzvah attack `http://www.crypto.com/papers/others/rc4_ksaproc.pdf`
- POODLE `https://www.openssl.org/~bodo/ssl-poodle.pdf`
- Dancing protocols, POODLEs and other tales from TLS `https://blog.hboeck.de/archives/858-Dancing-protocols,-POODLEs-and-other-tales-from-TLS.html`
- BERserk `http://www.intelsecurity.com/advanced-threat-research/berserk.html`

Sources VI

- BERserk PoC `https://github.com/FiloSottile/BERserk`
- Bleichenbacher Signature Forgery 2006 https://www.ietf.org/mail-archive/web/openpgp/current/msg00999.html
- miTLS - formally verified `http://www.mitls.org/`
- ocaml-tls `https://github.com/mirleft/ocaml-tls`
- Quote on gmail TLS performance `https://www.imperialviolet.org/2010/06/25/overclocking-ssl.html`
- SSL Strip `http://www.thoughtcrime.org/software/sslstrip/`

CC 0 ZERO

Sources VII

- HSTS Preload `https://hstspreload.appspot.com/`
- Bypassing HTTP Strict Transport Security `https://www.blackhat.com/docs/eu-14/materials/eu-14-Selvi-Bypassing-HTTP-Strict-Transport-Security-w pdf`
- Delorean NTP MitM `https://github.com/PentesterES/Delorean`
- Ring Learning With Errors / post-quantum key exchange `http://www.douglas.stebila.ca/research/papers/bcns15`
- SPHINCS / post quantum signatures `http://sphincs.cr.yp.to/`

CC 0 ZERO

Sources VIII

- Qualys SSL Labs Test
  `https://www.ssllabs.com/ssltest/`

CC 0 ZERO