# Transport Layer Security

## 194.144 Privacy Enhancing Technologies

Dr. Wilfried Mayer

# Outline 1/2

- Overview
- TLS Protocols
- PKI
- Implementations
- Cryptographic primitives

# Outline 2/2

- Application of TLS
- HTTPS
- User Behaviour
- Incidents, Flaws and Attacks
- Improvements
- TLS 1.3

# Overview

# Goals of TLS[1]

- Authentication
- Confidentiality
- Integrity
- TLS is application protocol independent

[1] RFC8446

# Goals of TLS 1.2[3]

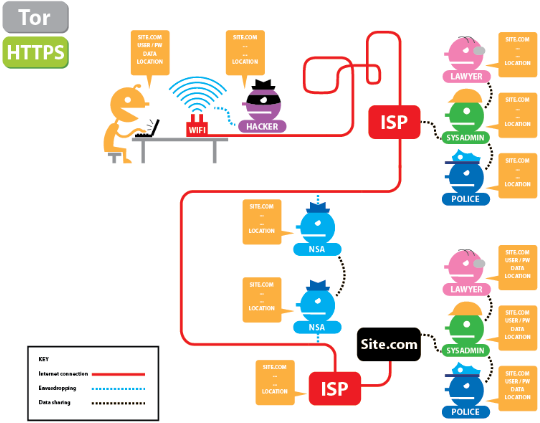
- Cryptographic security
- Interoperability
- Extensibility
- Relative efficiency[2]

[2] https://istlsfastyet.com/

[3] RFC5246

# TLS / PETS

- **Foundation of encrypted internet**
- Improvements / Incidents / Vulnerabilites
- Metadata not private
- No silver bullet for security


Tor
HTTPS
SITE.COM USER / PW DATA LOCATION
SITE.COM ... ... LOCATION
WIFI
HACKER
ISP
LAWYER
SYSADMIN
POLICE
SITE.COM ... ... LOCATION
NSA
NSA
Site.com
ISP
SITE.COM USER / PW DATA LOCATION
KEY
Internet connection
Eavesdropping
Data sharing

Interactive Overview (EFF)

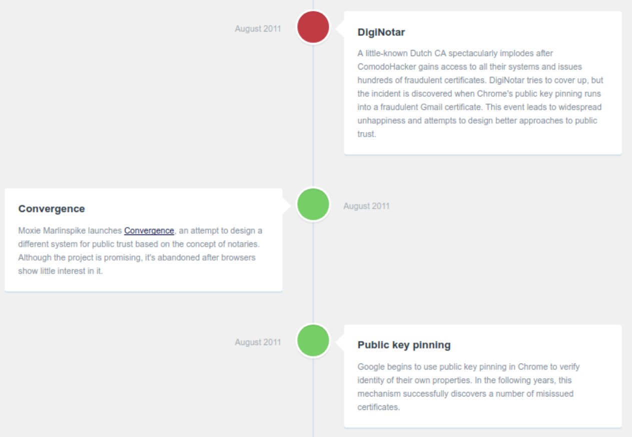
# Timeline

- SSL 2 by Netscape (1994)
- SSL 3 (1995)
- TLS 1.0 (1999, RFC 2246)
- TLS 1.1 (2006, RFC 4346)
- TLS 1.2 (2008, RFC 5246)
- TLS 1.3 (2018, RFC 8446 (after 28 drafts))

# Name

*Secure Sockets Layer* vs. **Transport Layer Security**

# Interactive SSL/TLS History



August 2011

**DigiNotar**

A little-known Dutch CA spectacularly implodes after ComodoHacker gains access to all their systems and issues hundreds of fraudulent certificates. DigiNotar tries to cover up, but the incident is discovered when Chrome's public key pinning runs into a fraudulent Gmail certificate. This event leads to widespread unhappiness and attempts to design better approaches to public trust.

**Convergence**

Moxie Marlinspike launches Convergence, an attempt to design a different system for public trust based on the concept of notaries. Although the project is promising, it's abandoned after browsers show little interest in it.

August 2011

August 2011

**Public key pinning**

Google begins to use public key pinning in Chrome to verify identity of their own properties. In the following years, this mechanism successfully discovers a number of misissued certificates.

See at feistyduck.com

# TLS Protocols

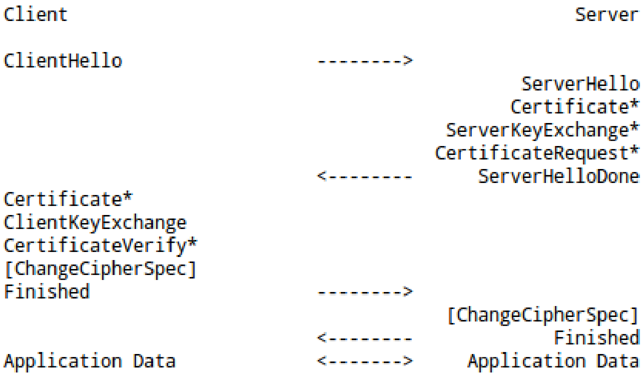
# RFC complexity

RFC5246, RFC2246, RFC4346, RFC6101, RFC2595, RFC2712, RFC2817, RFC2818, RFC3207, RFC3268, RFC3546, RFC3749, RFC3943, RFC4132, RFC4162, RFC4217, RFC4279, RFC4347, RFC4366, RFC4492, RFC4680, RFC4681, RFC4785, RFC5054, RFC5077, RFC5081, RFC5288, RFC5289, RFC5746, RFC5878, RFC5932, RFC6066, RFC6091, RFC6176, RFC6209, RFC6347, RFC6367, RFC6460, RFC6655, RFC7027, RFC7251, RFC7301, RFC7366, RFC7465, RFC7507, RFC7568, RFC7627, RFC7685, RFC5216, RFC7457, RFC7525, RFC8446, ...

# Two primary concepts

- Handshake protocol
  - authenticates the communicating parties
  - negotiates cryptographic modes
  - establishes shared keying material
- Record protocol
  - protect traffic between the communicating peers

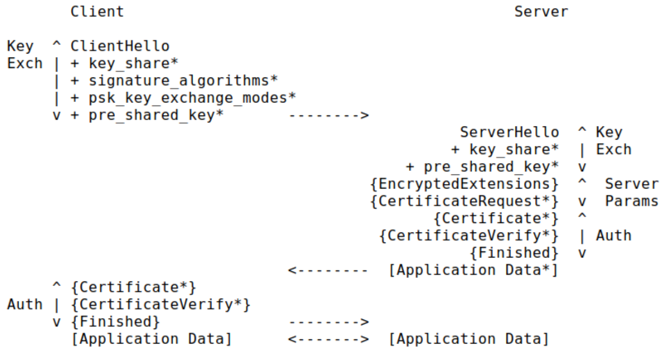
# Full Handshake TLS 1.2



```
Client                                               Server

ClientHello                  -------->
                                                ServerHello
                                               Certificate*
                                         ServerKeyExchange*
                                        CertificateRequest*
                             <--------      ServerHelloDone
Certificate*
ClientKeyExchange
CertificateVerify*
[ChangeCipherSpec]
Finished                     -------->
                                         [ChangeCipherSpec]
                             <--------             Finished
Application Data             <------->     Application Data
```

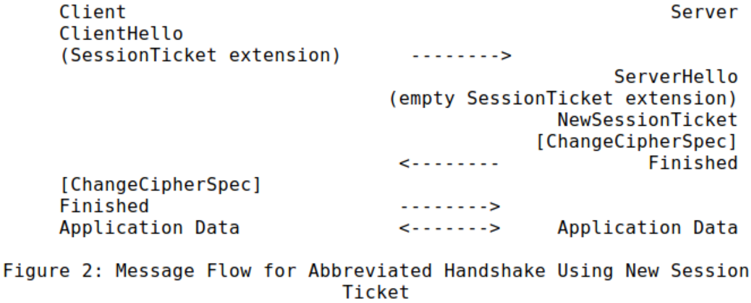
# Full Handshake TLS 1.3



```
       Client                                           Server

Key  ^ ClientHello
Exch | + key_share*
     | + signature_algorithms*
     | + psk_key_exchange_modes*
     v + pre_shared_key*       -------->
                                                  ServerHello  ^ Key
                                                 + key_share*  | Exch
                                            + pre_shared_key*  v
                                        {EncryptedExtensions}  ^  Server
                                        {CertificateRequest*}  v  Params
                                               {Certificate*}  ^
                                         {CertificateVerify*}  | Auth
                                                   {Finished}  v
                               <--------  [Application Data*]
     ^ {Certificate*}
Auth | {CertificateVerify*}
     v {Finished}              -------->
       [Application Data]      <------->  [Application Data]
```

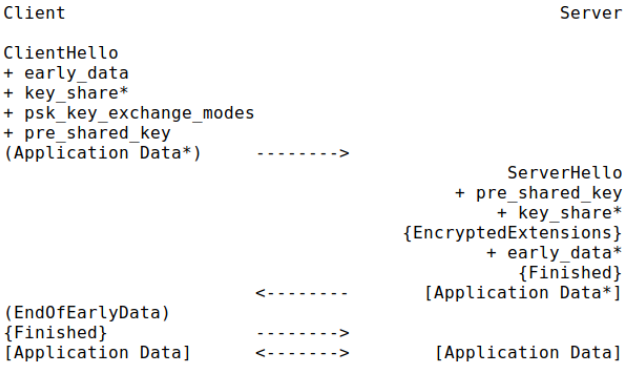
# Handshake Resumption TLS 1.2



```
Client                                                Server
ClientHello
(SessionTicket extension)      -------->
                                                 ServerHello
                             (empty SessionTicket extension)
                                            NewSessionTicket
                                          [ChangeCipherSpec]
                              <--------             Finished
[ChangeCipherSpec]
Finished                      -------->
Application Data              <------->     Application Data
```

Figure 2: Message Flow for Abbreviated Handshake Using New Session Ticket

# 0-RTT Handshake TLS 1.3



```
Client                                                Server

ClientHello
+ early_data
+ key_share*
+ psk_key_exchange_modes
+ pre_shared_key
(Application Data*)     -------->
                                                 ServerHello
                                            + pre_shared_key
                                                + key_share*
                                       {EncryptedExtensions}
                                               + early_data*
                                                  {Finished}
                        <--------        [Application Data*]
(EndOfEarlyData)
{Finished}              -------->
[Application Data]      <------->         [Application Data]
```

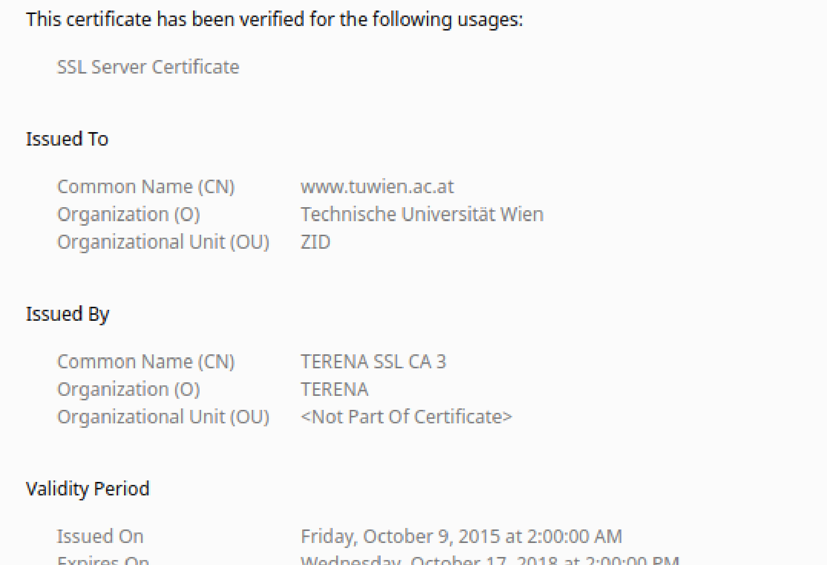
Security properties for 0-RTT data are weaker!

# PKI

# Public Key Infrastructure

- Certificates based on Pubkey encryption
- CA (Certificate Authority) issues certificates
- CA rights can be delegated: Sub-CAs
- Chain of trust (Chain of certificates) to the root CAs
- Root CAs are trusted

# Certificate



This certificate has been verified for the following usages:

SSL Server Certificate

Issued To

| | |
|---|---|
| Common Name (CN) | www.tuwien.ac.at |
| Organization (O) | Technische Universität Wien |
| Organizational Unit (OU) | ZID |

Issued By

| | |
|---|---|
| Common Name (CN) | TERENA SSL CA 3 |
| Organization (O) | TERENA |
| Organizational Unit (OU) | <Not Part Of Certificate> |

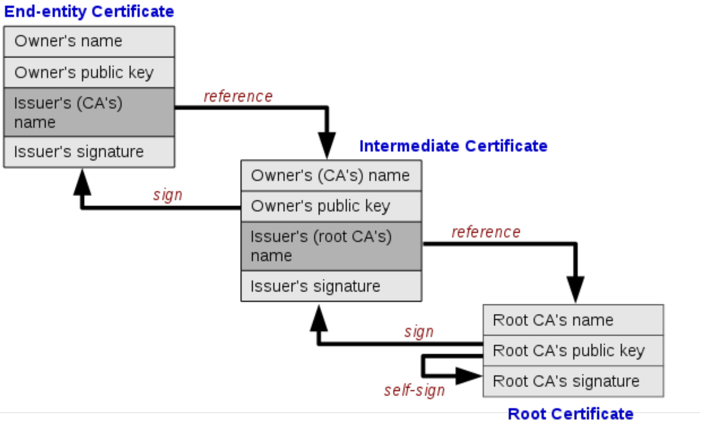
Validity Period

| | |
|---|---|
| Issued On | Friday, October 9, 2015 at 2:00:00 AM |
| Expires On | Wednesday, October 17, 2018 at 2:00:00 PM |

# X.509

- Standard for pubkey certificates
- Structured, e.g.:
  - Issuer Name
  - Subject name (inkl. Common Name)
  - Validity period
  - Extensions
  - ...
- .pem / .crt / .cer / .der / not .csr / not .key / ...

# Chain of trust



End-entity Certificate

- Owner's name
- Owner's public key
- Issuer's (CA's) name
- Issuer's signature

Intermediate Certificate

- Owner's (CA's) name
- Owner's public key
- Issuer's (root CA's) name
- Issuer's signature

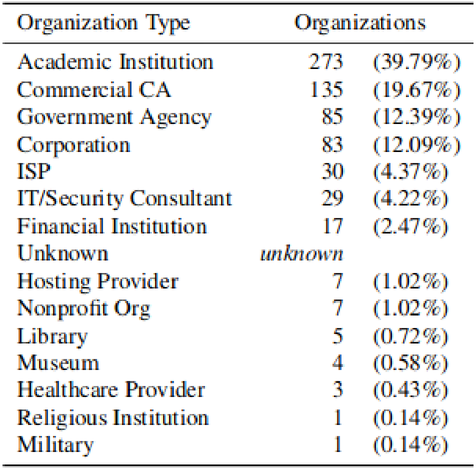
- Root CA's name
- Root CA's public key
- Root CA's signature

Root Certificate

# Root CAs, Trust stores

- Each browser and operating system has its set of trusted CAs
- These CAs could sign everything
- Not all signed HTTPS Certificates
- Controlled by different organizations, nations, ...
- Three organizations controlling 75% of trusted certificates

# Root CAs



| Organization Type | Organizations | |
|---|---|---|
| Academic Institution | 273 | (39.79%) |
| Commercial CA | 135 | (19.67%) |
| Government Agency | 85 | (12.39%) |
| Corporation | 83 | (12.09%) |
| ISP | 30 | (4.37%) |
| IT/Security Consultant | 29 | (4.22%) |
| Financial Institution | 17 | (2.47%) |
| Unknown | *unknown* | |
| Hosting Provider | 7 | (1.02%) |
| Nonprofit Org | 7 | (1.02%) |
| Library | 5 | (0.72%) |
| Museum | 4 | (0.58%) |
| Healthcare Provider | 3 | (0.43%) |
| Religious Institution | 1 | (0.14%) |
| Military | 1 | (0.14%) |

Durumeric et al., Analysis of the HTTPS Certificate Ecosystem

# CA/Browser Forum

- Defines Baseline Requirements
- Rules that CAs have to follow
- https://cabforum.org/baseline-requirements-documents/

# Implementation

# Implementations

- OpenSSL: de-facto standard, swiss army knife
- LibreSSL: fork by the OpenBSD team
- BoringSSL: fork by Google
- GnuTLS: initial GNU implementation
- NSS: by Mozilla
- Microsoft Secure Channel
- s2n: implementation by Amazon
- miTLS: Verified Implementation
- See full comparison here

# OpenSSL problems

- Had its own memory management which prevented many analysis tools
- Bugs unfixed for a long time
- Code base completely unreadable
- Extensive backward compatibility

# OpenSSL vs. LibreSSL

- BSD team forked OpenBSD 1.0.1g after Heartbleed
- 90.000 LOC deleted within 30 days (initially 388.000)
- Part of OpenBSD now

Google forked towards BoringSSL

# Cryptographic primitives

# Ciphersuites until 1.2

- Remember: Extensibility
- Specifies cryptographic algorithms and modes

Ciphersuites consist of:

- Key exchange
- Authentication
- Symmetric cryptography for transport
- Integrity (Hash)

# Ciphersuites until 1.2

TLS_DHE_RSA_WITH_AES_256_CBC_SHA

- DHE for key exchange
- RSA for authentication
- AES 256bit in CBC mode for encryption
- SHA for hashing

IANA:

- More than 500 defined[4]
- Two bytes define the ciphersuite

[4] Source: list by IANA

# Ciphersuites until 1.2

Key exchange (selection):

- RSA for authentication
- Problem with RSA: private key can decrypt previous communication content

Forward secrecy:

- DHE_RSA: ephemeral Diffie-Hellman
- ECDHE_RSA: elliptic curve Diffie-Hellman

# Ciphersuites until 1.2

Encryption:

- Block or stream ciphers
- Block: AES, 3DES, Camellia
- Stream: RC4, ChaCha

# Ciphersuites until 1.2

- Server supports a certain set
- Browser supports a certain set
- Negotiated while Handshake

# Ciphersuites for TLS 1.3

- Highly reduced set (5)!
- Not compatible with TLS 1.2
- All support Forward Secrecy
- Authenticated Encryption with Associated Data (AEAD)

# Which Ciphersuites to use?

- e.g., recommendations by Mozilla
- Recommended configurations
- Mozilla SSL Configuration Generator

# Application of TLS

# HTTPS

- Most widely used application layer protocol for TLS
- HTTP over 443
- You all use it!

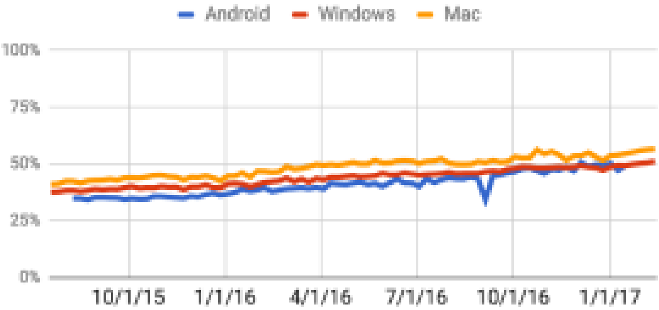
# HTTPS Problems

- HTTPS Adoption
- Secure Deployment
- Usability
- Who leads the way?

# HTTPS Adoption

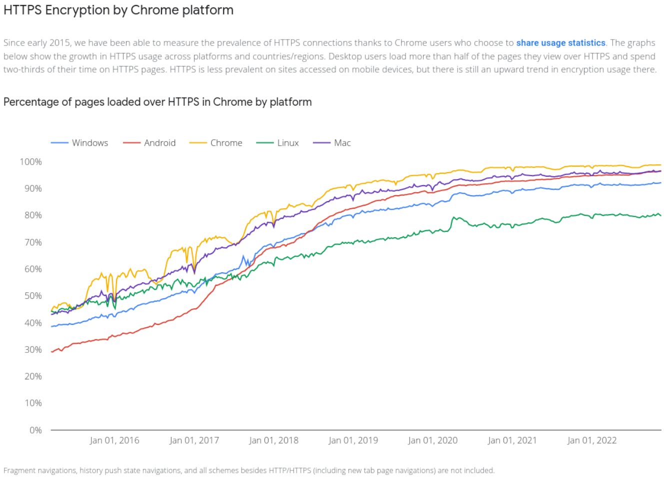
- HTTPS was not used widely enough
- HTTPS used only for “high important” pages
- Certificates cost money (pre Let’s Encrypt era)
- Self-signed certificates bring problems

# HTTPS Adoption



Felt et al., Measuring HTTPS Adoption on the Web

# HTTPS Adoption 2022



HTTPS Encryption by Chrome platform

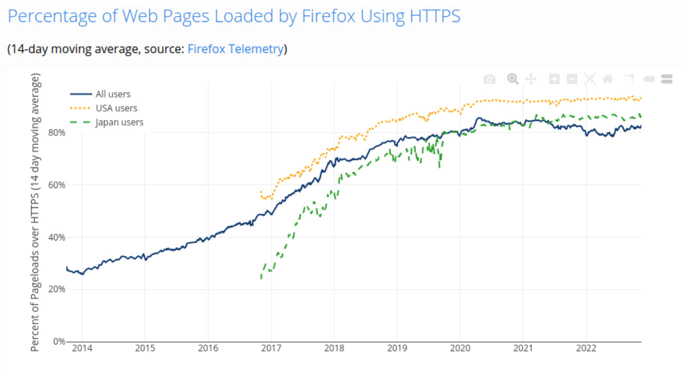
Since early 2015, we have been able to measure the prevalence of HTTPS connections thanks to Chrome users who choose to **share usage statistics**. The graphs below show the growth in HTTPS usage across platforms and countries/regions. Desktop users load more than half of the pages they view over HTTPS and spend two-thirds of their time on HTTPS pages. HTTPS is less prevalent on sites accessed on mobile devices, but there is still an upward trend in encryption usage there.

Percentage of pages loaded over HTTPS in Chrome by platform

Fragment navigations, history push state navigations, and all schemes besides HTTP/HTTPS (including new tab page navigations) are not included.

https://transparencyreport.google.com/https/overview

# HTTPS Adoption 2022

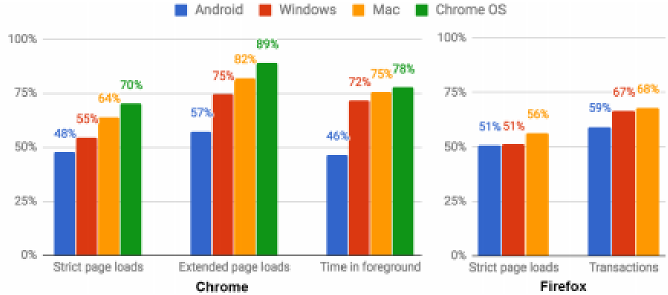


Percentage of Web Pages Loaded by Firefox Using HTTPS

(14-day moving average, source: Firefox Telemetry)

https://letsencrypt.org/stats/

# HTTPS Adoption



Felt et al., Measuring HTTPS Adoption on the Web

# HTTPS Adoption - Browser Warnings

- 2016/10 Warning for unencrypted form data
- 2018/07 Warning for HTTP
- 2020/10 Chrome 86 forbids mixed content
- 2021/04 Chrome 90 defaults to HTTPS

# Secure Deployment

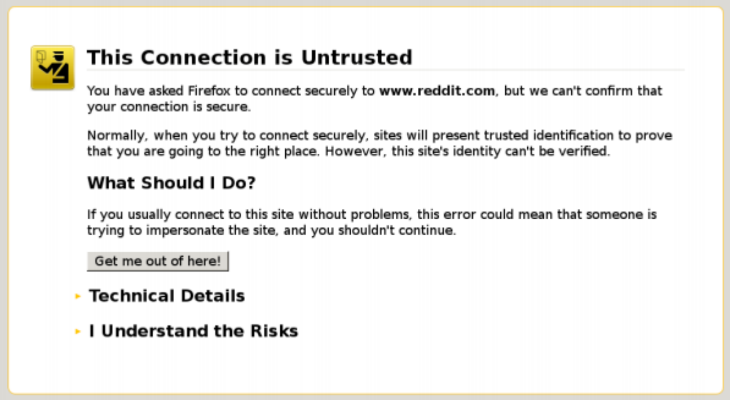
- Complex task
- How to do correctly?
- What is a “secure” deployment?
- e.g., correct Ciphersuites (see above)
- Grading with SSLTest (see below)

# HTTPS Usability

- Security for people (TUW mission: Technology for people)
- Disruptive Security Concepts (Browser Warnings)
- Connection Security Indicators (Browser Icons)
- Admins should be seen as users too
- *Given a choice between dancing pigs and security, the user will pick dancing pigs every time*[5]

[5] Felten and McGraw

# Browser Warnings



## This Connection is Untrusted

You have asked Firefox to connect securely to **www.reddit.com**, but we can't confirm that your connection is secure.

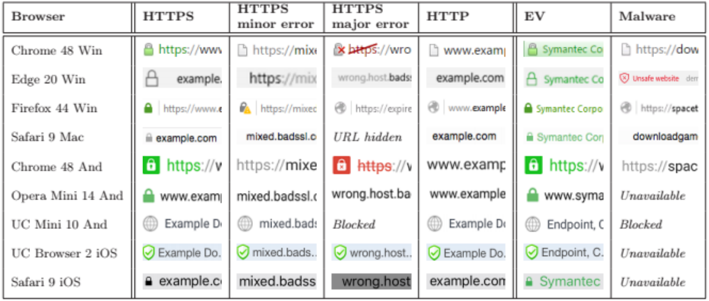
Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

### What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Akhawe, Felt: Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness

# Connection Security Indicators

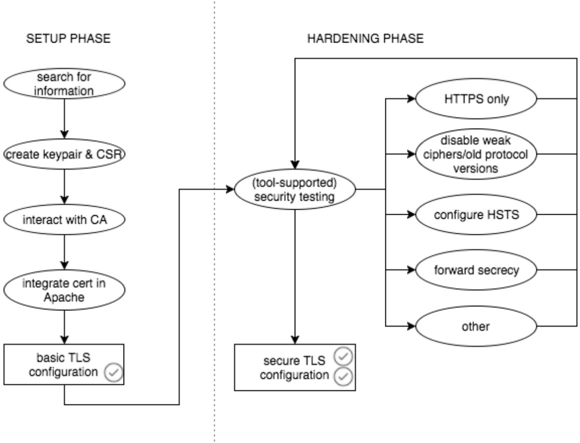


| Browser | HTTPS | HTTPS minor error | HTTPS major error | HTTP | EV | Malware |
|---|---|---|---|---|---|---|
| Chrome 48 Win | https://www | https://mixe | ~~https~~://wro | www.examp | Symantec Co | https://dow |
| Edge 20 Win | example. | https://mix | wrong.host.badss | example.com | Symantec Co | Unsafe website dem |
| Firefox 44 Win | https://www.e | https://mixed | https://expire | www.example | Symantec Corpo | https://spacet |
| Safari 9 Mac | example.com | mixed.badssl.c | *URL hidden* | example.com | Symantec Corp | downloadgam |
| Chrome 48 And | https://w | https://mixe | https://w | www.examp | https://w | https://spac |
| Opera Mini 14 And | www.examp | mixed.badssl.c | wrong.host.ba | www.example | www.syma | *Unavailable* |
| UC Mini 10 And | Example Do | mixed.bads | *Blocked* | Example Do | Endpoint, C | *Blocked* |
| UC Browser 2 iOS | Example Do.. | mixed.bads.. | wrong.host.. | Example Do.. | Endpoint, C.. | *Unavailable* |
| Safari 9 iOS | example.c | mixed.badss | wrong.host | example.com | Symantec | *Unavailable* |

**Figure 2: Security indicators for major browsers on Windows (Win), Mac, Android (And), and iOS. For categories that trigger warnings (e.g., malware), we include the security indicator state during the warning.**

Felt et al., Rethinking Connection Security Indicators

# TLS Deployment process



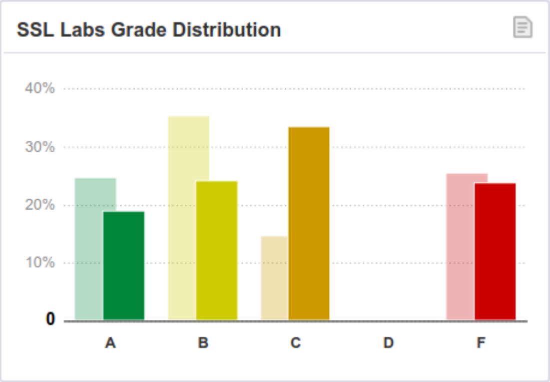
Krombholz, Mayer, Schmiedecker, Weippl: ”I Have No Idea What I’m Doing” - On the Usability of Deploying HTTPS

# Deployments

- Hard to find a good configuration
- No secure defaults
- Bad documentation
- Lacking tool support
- Situation is constantly improved

# TLS Deployment

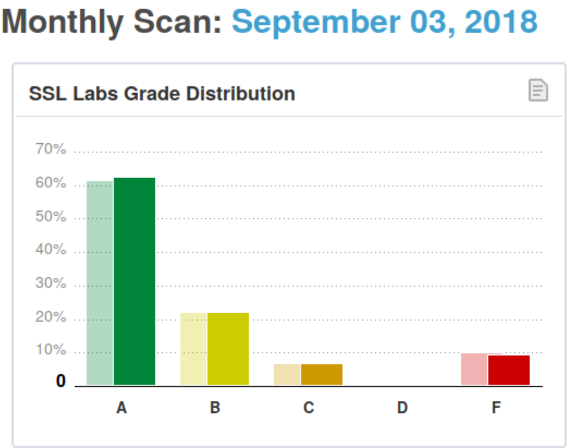
## Monthly Scan: June 11, 2015



SSL Labs Grade Distribution

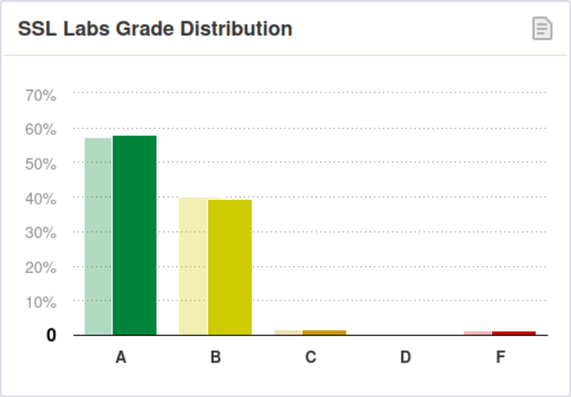
SSLPulse

# TLS Deployment



SSLPulse

# TLS Deployment



SSLPulse

# Who leads the way?

- Browsers, CAs, Service provider
  - Gmail HTTPS by default since January 2010
  - Google Search (if users logged in) since 2011
  - Forward secrecy since November 2011
  - Facebook for all since July 2013
  - Google Search for all since Sept. 2013
  - Let’s Encrypt 2015
  - Google+Facebook warning for HTTP passwords 2017
  - Google requires CT in April 2018

# TLS for Email

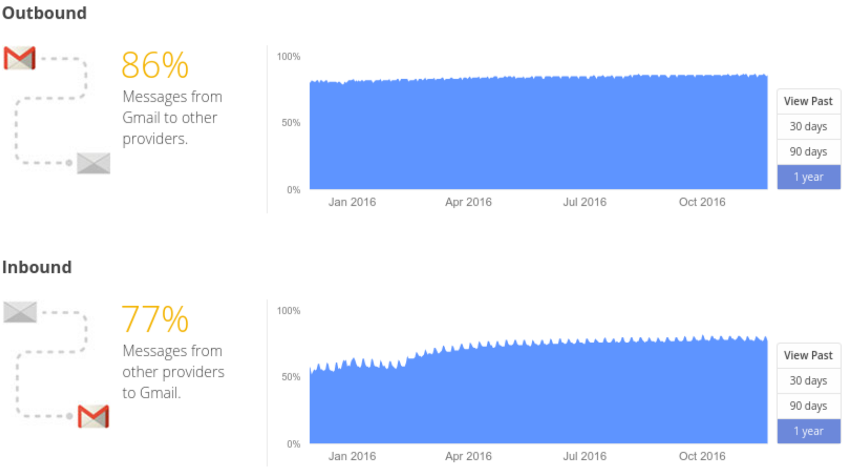
- Dedicated TLS ports (465, 993, 995)
- STARTTLS to upgrade unencrypted connections
    - Important for all email protocols: POP, IMAP, SMTP (110, 143, 25)
    - “Opportunistic encryption” - if possible
    - Does not defeat active attackers

# TLS for Email

Differences for SMTP vs. POP/IMAP:

- How do two parties verify certificates?
- All extensions from HTTPS not applicable
- No user warnings (lock icon), ...

# STARTTLS Transparency



**Outbound**

86%

Messages from Gmail to other providers.

**Inbound**

77%

Messages from other providers to Gmail.

# Incidents, Attacks and Flaws

# Incidents, Attacks and Flaws

- Remember:
    - Protocol
    - PKI
    - Implementations
    - Cryptographic primitives
    - Applications (HTTPS, Email)
    - User Behaviour
- add Murphy’s law

# Incidents PKI: DigiNotar

- CA from the Netherlands, hacked in July 2011
- Fox-IT investigated the attack
- DigiNotar went bankrupt, was removed from all browsern in August 2011

# Incidents PKI: DigiNotar

Problems:

- All signing servers were in one AD, weak password
- Reachable over the management LAN
- No antivirus on the servers
- Public webserver was unpatched

# Incidents PKI

*.*.com
*.android.com
*.balatarin.com (3)
*.globalsign.com (7)
*.logmein.com
*.mozilla.org
*.skype.com (22)
*.torproject.org (14)
*.wordpress.com (14)
Comodo Root CA (20)
Equifax Root CA (40)
login.live.com (17)
secure.logmein.com (17)
VeriSign Root CA (21)
www.balatarin.com (16)
www.Equifax.com
www.google.com (12)
www.sis.gov.uk (10)

*.*.org
*.aol.com
*.comodo.com (3)
*.google.com (26)
*.microsoft.com (3)
*.RamzShekaneBozorg.com
*.startssl.com
*.walla.co.il (2)
addons.mozilla.org (17)
CyberTrust Root CA (20)
friends.walla.co.il (8)
login.yahoo.com (19)
Thawte Root CA (45)
wordpress.com (12)
www.cia.gov (25)
www.facebook.com (14)
www.hamdami.com
www.update.microsoft.com (4)

*.10million.org (2)
*.azadegi.com (2)
*.digicert.com (2)
*.JanamFadayeRahbar.com
*.mossad.gov.il (2)
*.SahebeDonyayeDigital.com
*.thawte.com (6)
*.windowsupdate.com (3)
azadegi.com (16)
DigiCert Root CA (21)
GlobalSign Root CA (20)
my.screenname.aol.com
twitter.com (18)
www.10million.org (8)
www.cybertrust.com
www.globalsign.com
www.mossad.gov.il (5)

# Incidents PKI: DigiNotar

Operation Black Tulip:

- Detected due to TLS pinning in Chrome
- At least 531 fraudulent certificates were issued
- Visualization using OCSP requests[6]
- Used to attack Gmail users MITM in Iran (95% of all the OCSP requests)

[6] Link: Video

# Incidents PKI: TLS Incidents

Sometimes using sub-CA issuer and certificates:

- Turktrust: December 2012
    - Sub-CA certificate which was deployed in firewalls
- ANSSI: December 2013,
    - French sub-CA issuing certificates for Google
- India: July 2014
    - Indian sub-CA got hacked

# Incidents PKI: CAs distrusted

- October 2016: Apple, Chrome, and Mozilla distrust WoSign and StartCom
- … multiple rule violations …
- September 2017: Google and Mozilla decide to stop trusting existing Symantec certificates
- Announcement in March
- …Despite having knowledge of these issues, Symantec has repeatedly failed to proactively disclose them.…

# Incidents PKI

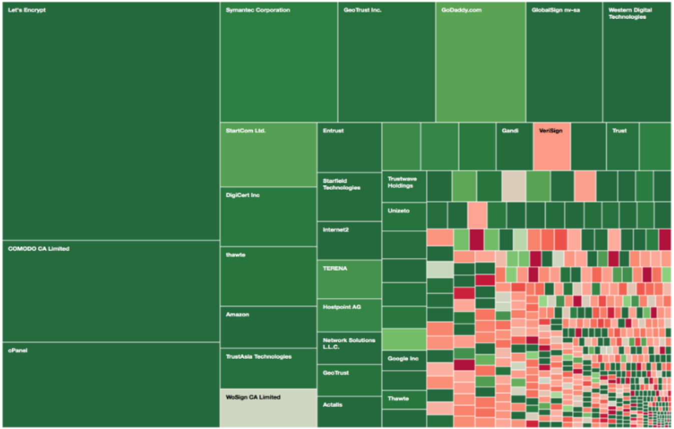
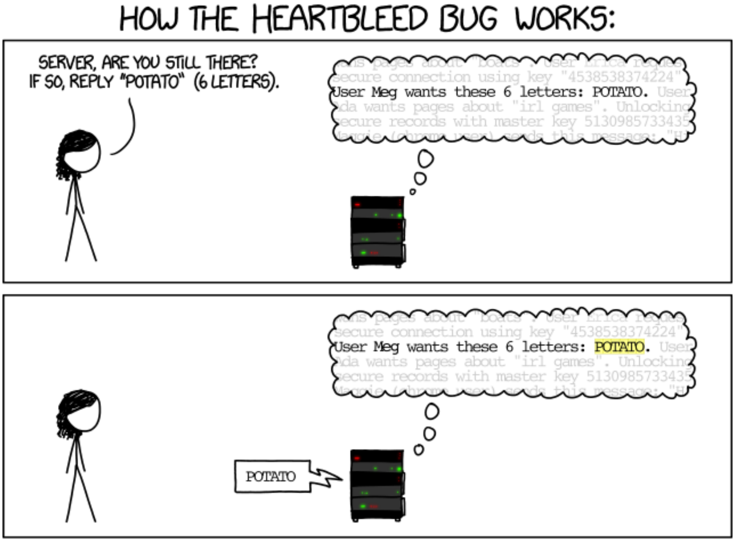


Fig. 5: **Percent ZLint Errors by Total Certificates Issued**—Large certificate authorities generally issue certificates with fewer ZLint errors than smaller authorities.

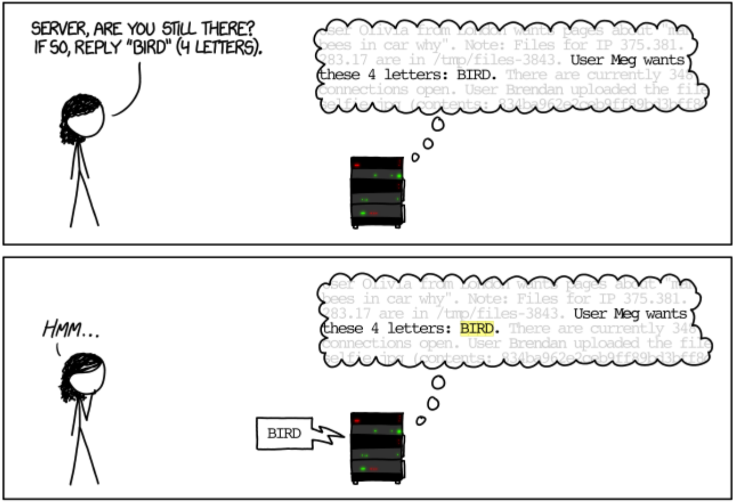
Kumar et al., Tracking Certificate Misissuance in the Wild

# Implementation bug: Heartbleed

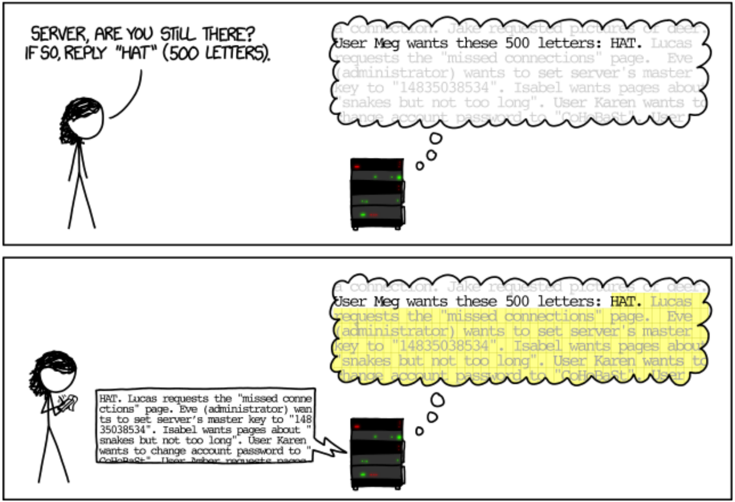
- Vulnerability in OpenSSL, April 2014
- In the Heartbeat protocol in TLS, missing bounds check
- Up to 64kb readable from the heap
- Could contain user data, passwords and TLS private key


HOW THE HEARTBLEED BUG WORKS:
SERVER, ARE YOU STILL THERE? IF SO, REPLY "POTATO" (6 LETTERS).
ants pages about "boats". User Erica reques
secure connection using key "4538538374224"
User Meg wants these 6 letters: POTATO. User
Ada wants pages about "irl games". Unlocking
ecure records with master key 5130985733435
Maggie (chrome user) sends this message: "H
ants pages about "boats". User Erica reques
secure connection using key "4538538374224"
User Meg wants these 6 letters: POTATO. User
Ada wants pages about "irl games". Unlocking
ecure records with master key 5130985733435
Maggie (chrome user) sends this message: "H
POTATO

Source: xkcd

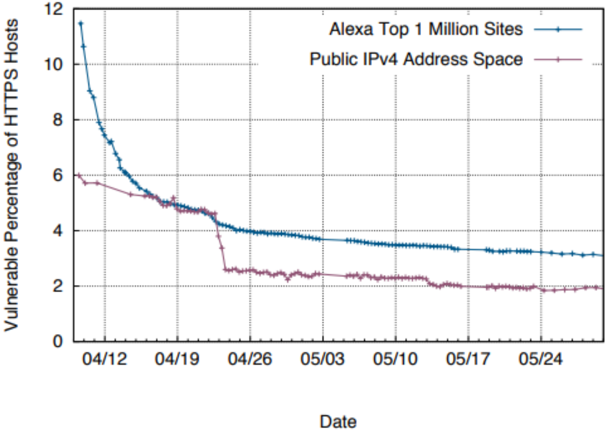

SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).
ser Olivia from London wants pages about "ma bees in car why". Note: Files for IP 375.381. 283.17 are in /tmp/files-3843. User Meg wants these 4 letters: BIRD. There are currently 348 connections open. User Brendan uploaded the file selfie.jpg (contents: 834ba962e2ceb9ff89bd3bff8
HMM...
ser Olivia from London wants pages about "ma bees in car why". Note: Files for IP 375.381. 283.17 are in /tmp/files-3843. User Meg wants these 4 letters: BIRD. There are currently 348 connections open. User Brendan uploaded the file selfie.jpg (contents: 834ba962e2ceb9ff89bd3bff8
BIRD

Source: xkcd


SERVER, ARE YOU STILL THERE? IF SO, REPLY "HAT" (500 LETTERS).
a connection. Jake requested pictures of deer. User Meg wants these 500 letters: HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoHoBaSt". User
a connection. Jake requested pictures of deer. User Meg wants these 500 letters: HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoHoBaSt". User
HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoHoBaSt". User Amber requests pages

Source: xkcd

# Heartbleed worldwide



Durumeric et al., The Matter of Heartbleed

# Crypto - Ps and Qs

- Problem for creating pubkeys
- RSA chooses parameter at random for pubkey
- For devices with low entropy collision possible
- In particular problematic for embedded devices
- 0.5% of all IPv4 in 2012

Heninger et al., Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices

# Crypto - Other Flaws

- Attacks against RC4
    - Considered a work around after BEAST, 2013
    - Insecure, Break RC4 feasable
    - Forbidden by RFC, Feb. 2015
- Debian Weak Keys
    - Weak Random Number Generator 2005
    - Weak keys

# Protocol Flaws

- DROWN
  - Decrypting RSA with Obsolete and Weakened eNcryptio
  - Exploits support of SSLv2
  - Certificate on other server with SSLv2?
- POODLE
  - Padding Oracle On Downgraded Legacy Encryption
  - Design of SSLv3 - Padding oracle attack against CBC mode
  - Prevent downgrade with TLS_FALLBACK_SCSV

# Other TLS Attacks

- SMACK (State Machine AttaCKs)
- Logjam (Downgrade, Weak Diffie-Hellman)
- FREAK (Downgrade, Factoring RSA Export Keys)
- CRIME, BREACH (HTTP compression)
- Lucky 13 (cryptographic timing attack against CBC mode)
- STRIPTLS attack (opportunistic encryption - application)
- ....

# Improvements

# Why improving TLS?

You still ask?

# Improvements

- HSTS
- Certificate Pinning
- HPKP (dead)
- Certificate Transparency
- CAA
- Let’s Encrypt
- Tool Support

# HSTS

- HTTP Strict Transport Security
- Part of the HTTP Header response from the server
- Stores HTTPS preference
- `Strict-Transport-Security max-age=31536000`
- Error message instead of warning

# HSTS cont.

- Problem: TOFU (Trust On First Use)
- Preload list
- Firefox and Chrome HSTS preload list[7]

[7] List of URLS: https://www.chromium.org/hsts

# Pinning

- Key Distribution Problem
- ”Solved” with PKI, but PKI has it’s problems
- Pin the certificate or public key
- e.g., directly in browser or source code
- not scalable

# HPKP

- No Support! Dead idea...
- HTTP Public Key Pinning
- Part of the HTTP Header response from the server
- Stores Pinned Key

-
  ```
  Public-Key-Pins:
  pin-sha256="cUPcTAZWKaASuYWhhneDttWpY3oBAkE3h2+soZS
  pin-sha256="M8HztCzM3elUxkcjR2S5P4hhyBNf6lHkmjAHKhp
  max-age=5184000; includeSubDomains;
  report-uri="https://www.example.org/hpkp-report"
  ```

# HPKP dead?

- Pin: Leaf cert, Intermediate cert or Root cert
- Public-Key-Pins-Report-Only
- Is HTTP Public Key Pinning Dead?[8]
- Fixing HPKP with Pin Revocation[9]
- Dead... Planned removal in Chrome 67, May 2018[10]

[8] https://blog.qualys.com/ssllabs/2016/09/06/is-http-public-key-pinning-dead

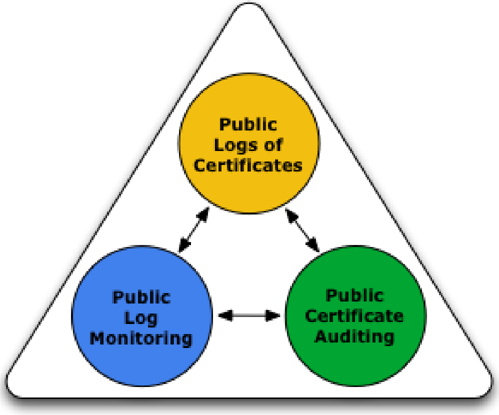
[9] https://blog.qualys.com/ssllabs/2017/09/05/fixing-hpkp-with-pin-revocation

[10] https://groups.google.com/a/chromium.org/forum/#!msg/blink-dev/he9tr7p3rZ8/eNMwKPmUBAAJ?hn

# CAA

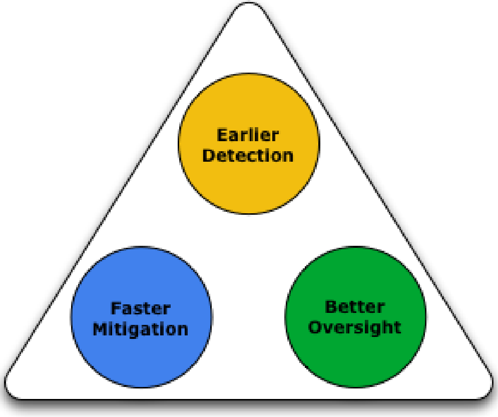
- DNS record: Certification Authority Authorization
- ”Which CAs are allowed to issue a certificate for my domain?”
- `example.com.  IN CAA 0 issue "letsencrypt.org"`
- Mandatory for CAs from September 2017
- CA check - not client system check

# Certificate Transparency



Source: Certificate Transparency

# Certificate Transparency



Source: Certificate Transparency

# Certificate Transparency

- RFC6962
- Logs: records of certificates
- Logs: everyone could host, but currently Google and CAs
- Monitor: watch for suspicious certificates
- Auditor: verify that logs are behaving correctly
- Warning for Certificates without CT Log Entry

Let’s
Encrypt

# Let’s Encrypt

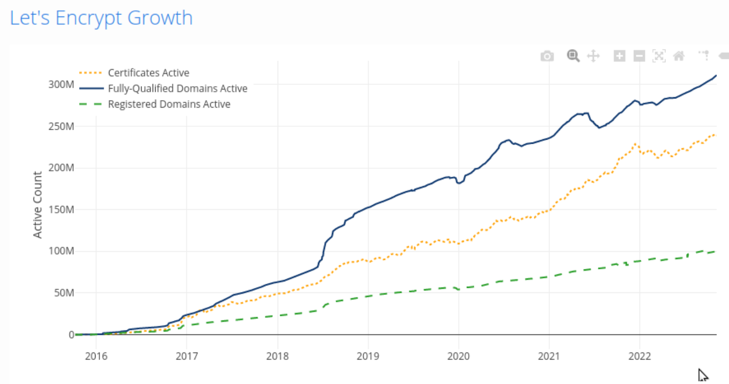
- Free CA!
- Open CA!
- Automated CA (Domain-based validation)!

# Let’s Encrypt

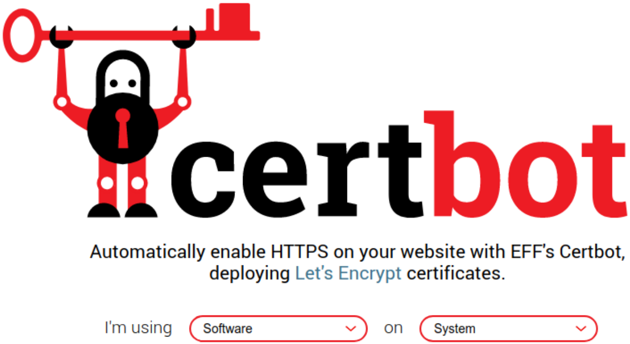
- ACME protocol in background[11]
- Easy TLS setup:
- “sudo apt-get install letsencrypt; letsencrypt run”
- Issued 100 million certs in June 2017

[11] https://github.com/letsencrypt/acme-spec

# Active Let's Encrypt certificates



# Certbot

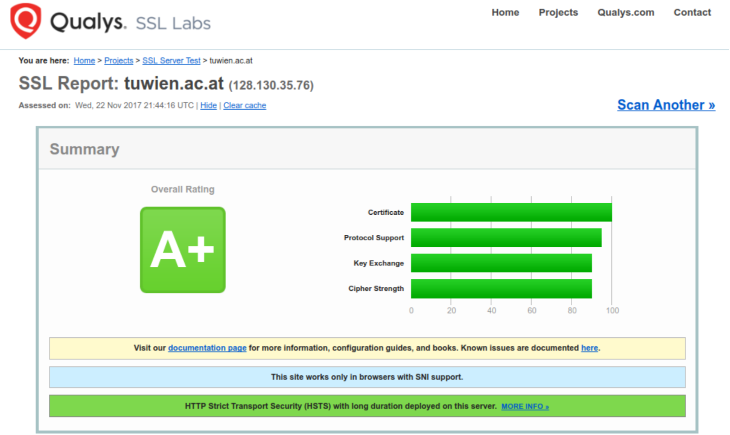


Automatically enable HTTPS on your website with EFF's Certbot, deploying Let's Encrypt certificates.

I'm using Software on System

Certbot EFF

# SSL Test



You are here: Home > Projects > SSL Server Test > tuwien.ac.at

SSL Report: tuwien.ac.at (128.130.35.76)

Assessed on: Wed, 22 Nov 2017 21:44:16 UTC | Hide | Clear cache

Scan Another »

Summary

Overall Rating

A+

Visit our documentation page for more information, configuration guides, and books. Known issues are documented here.

This site works only in browsers with SNI support.

HTTP Strict Transport Security (HSTS) with long duration deployed on this server. MORE INFO »

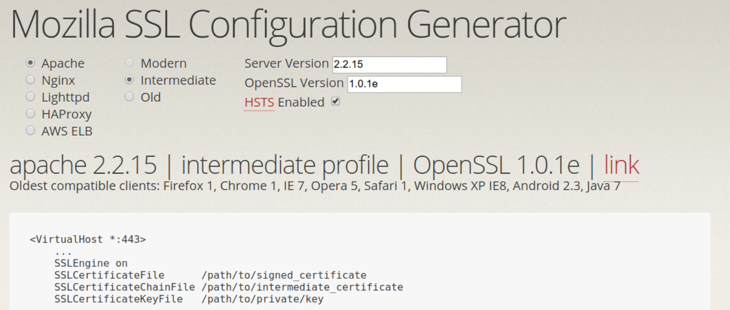
ssllabs ssltest

# Qualys Tools

SSL Pulse:

- Monthly scan for 200k top-websites
- Checks for complete certificate chain, CipherSuites, HSTS, ...
- But also attacks like CRIME, Beast, Heartbleed, ...
- Ranking according to Qualys SSL Labs

# Mozilla SSL Configuration Generator



Mozilla SSL Configuration Generator

● Apache
○ Nginx
○ Lighttpd
○ HAProxy
○ AWS ELB

○ Modern
● Intermediate
○ Old

Server Version: 2.2.15
OpenSSL Version: 1.0.1e
HSTS Enabled ☑

apache 2.2.15 | intermediate profile | OpenSSL 1.0.1e | link

Oldest compatible clients: Firefox 1, Chrome 1, IE 7, Opera 5, Safari 1, Windows XP IE8, Android 2.3, Java 7

```
<VirtualHost *:443>
    ...
    SSLEngine on
    SSLCertificateFile      /path/to/signed_certificate
    SSLCertificateChainFile /path/to/intermediate_certificate
    SSLCertificateKeyFile   /path/to/private/key
```

# HTTPS Everywhere

- Browser extension for Firefox & Chrome, by EFF
- Changes connections from HTTP to HTTPS (where available)
- Rule-based
- Manually maintained list
- link here

# DANE

- DNS-based Authentication of Named Entities
- Replace PKI and ask DNS
- Needs DNSSEC
- Not really used

# MTA-STS

- SMTP MTA Strict Transport Security
- ”HSTS for Email”
- RFC8461 (End of September 2018)

# TLS 1.3.

# Major differences

- Static RSA removed
- Forward Secrecy everywhere!
- CBC mode removed (Lucky13, Poodle, ...)
- Only AEAD (Authenticated Encryption with Associated Data) algorithms
- RC4, SHA1, MD5 removed
- Compression removed
- Renegotiation removed

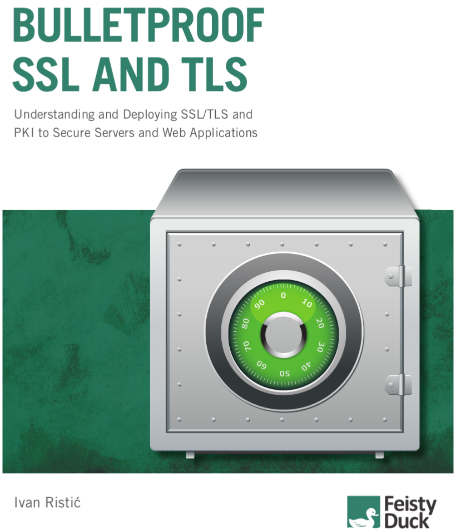
# Major differences

- Cipher suite concept changed
- Zero-RTT (Zero round-trip time (0-RTT) mode)
- Handshake state machine has been significantly restructured
- Fixed DHE groups (simplified)
- Session IDs + Tickets - Tickets + PSK
- Downgrade protection
- Full handshake signature

# Problems introducing TLS 1.3

- Middlebox decryption
- Version intolerance
    - TLS 1.3. (3.4. / 3.3.)
    - Make TLS 1.3. look like TLS 1.2.

# TLS Reference



BULLETPROOF SSL AND TLS

Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications

Ivan Ristić

Feisty Duck

# Questions?