Web-Security

HTTPS, OWASP

HTTPS
Hypertext Transfer Protocol Secure

HTTP ist grundsätzlich unverschlüsselt

- Jeder Mittelsmann kann daher die Kommunikation mitschneiden und verändern (MITM-Angriffe)
- Um sicherzugehen, dass die Inhalte vom Server stammen und unverändert ausgeliefert werden, muss HTTPS genutzt werden

HTTPS: HTTP over TLS/HTTP over SSL/HTTP Secure

- Baut wie HTTPS auf der Anwendungsschicht auf
- Sämtlicher Inhalt (Request, Reponse) wird verschlüsselt
- IP-Adresse und Port sind weiterhin für Mittelsmänner zu erkennen

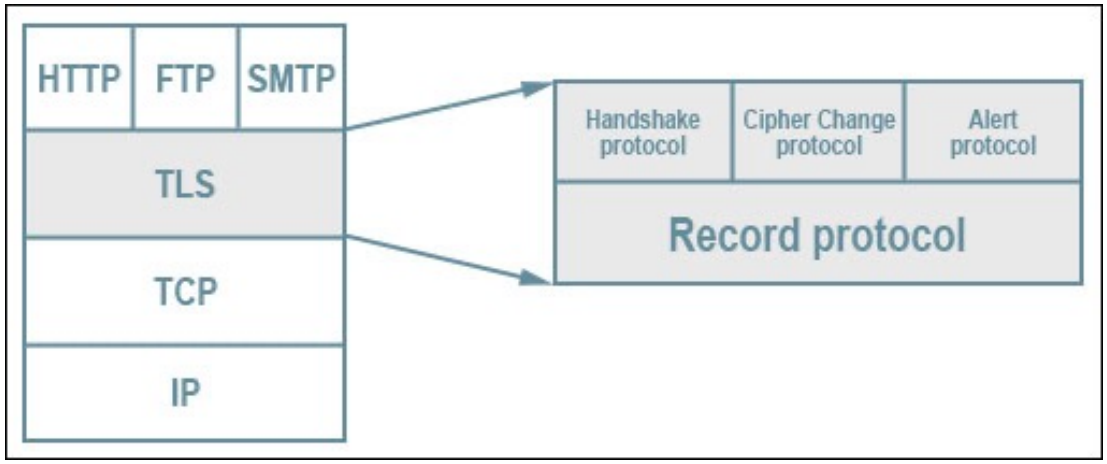
TLS/SSL

- TLS (Transport Layer Security) ist eine Weiterentwicklung von SSL (Secure Socket Layer) und wird heutzutage fast ausschließlich genutzt
- Die Begriffe *TLS* und *SSL* werden häufig durcheinander gebracht und oft synonym verwendet

HTTPS im RFC 2818 (2000): https://tools.ietf.org/html/rfc2818

HTTPS
TLS verschlüsselt die Anwendungsdaten

Transparente Schicht zwischen TCP/IP und Application Layer universell einsetzbar



https://commons.wikimedia.org/wiki/File:TLS_protocol_stack.jpg

HTTPS
Zertifikate

Verbindungsaufbau basiert auf asymmetrischer Verchlüsselung

- Hierzu ist ein Public und ein Private Key nötig
- Das Zertifikat entspricht hierbei dem Public Key, sowie zusätzlichen Meta-Informationen wie dem Servernamen, Gültigkeitszeitraum und Signaturen von anderen Zertifikaten

Verbindungsaufbau

- Verschlüsselt der Client Daten mit dem Public Key des Servers, so kann er sicher sein, dass auch nur der Besitzer des Public Keys die Daten entschlüsseln kann
- Der Client muss hierfür sicher wissen, dass es sich auch um den korrekten Public Key handelt

Datenaustausch basiert auf symmetrischer Verschlüsselung

- Client und Server einigen sich auf ein Secret mit dem die Übertragung der Daten passiert (da symmetrische Verschlüsselung schneller ist)

HTTPS
Verbindungsaufbau

Grundsätzliches Prinzip

- TLS Handshake (vereinfacht, mit Server-Authentifikation)
  1. Client meldet sich beim Server
  2. Server übermittelt sein Zertifikat inkl. Public Key (und Metadaten wie TLS Version)
  3. Client prüft das Server-Zertifikat (Public Key) und übermittelt ein Secret
     - Client stellt fest ob das Server-Zertifikat von einem vertrautem CA signiert ist und für die besuchte Webseite ausgestellt wurde
     - Client nutzt die Metadaten zur Auswahl des zu verwendenden Verschlüsselungsverfahrens
     - Client generiert ein Secret für die Kommunikation
     - Client verschlüsselt das Secret mit dem übermittelten Public Key und sendet es an den Server
  4. Server nutzt seinen Private Key um das Secret zu entschlüsseln und ist somit authentifiziert
  5. Beide Seiten sind nun im Besitz des Secrets, welches einem symmetrischem Sitzungsschlüssel entspricht; das konkrete Verfahren wurde vereinbart
  6. Die Kommunikation kann von hier an verschlüsselt fortgeführt werden
- Es gibt die Möglichkeit, den Handshake abzukürzen und sich auf einen schon ausgehandelten Sitzungsschlüssel zu beziehen
- Auch kann der Server ein Client-Zertifikat anfordern und dieses mit einer Challenge (Zufallszahl) validieren

Weitere Informationen: https://www.symantec.com/connect/blogs/how-does-ssl-work-what-ssl-handshake

HTTPS
Handshake kann dauern

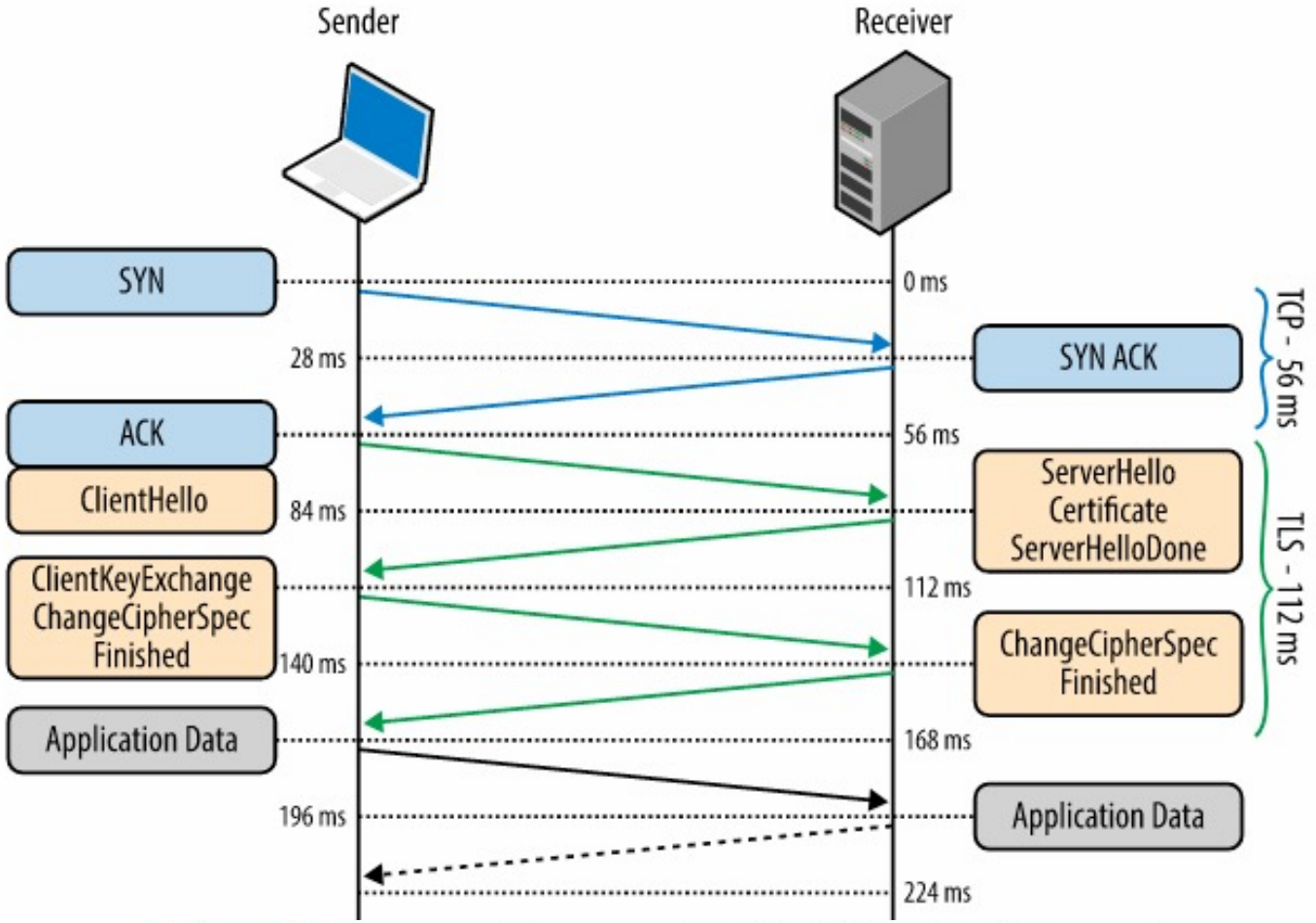
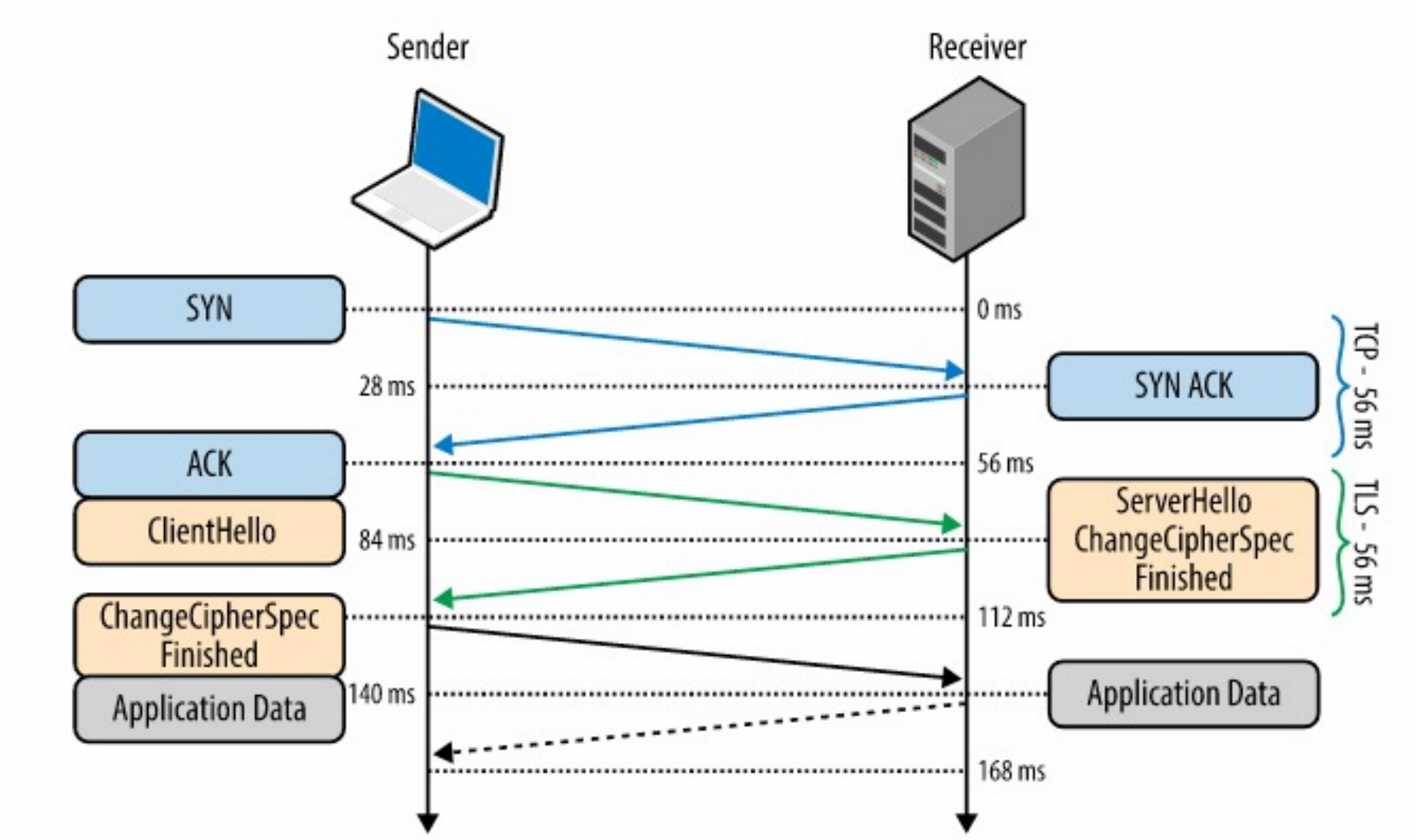


Figure 4-2 assumes the same (optimistic) 28 millisecond one-way "light in fiber" delay between New York and London as used in previous TCP connection establishment examples; see Table 1-1.

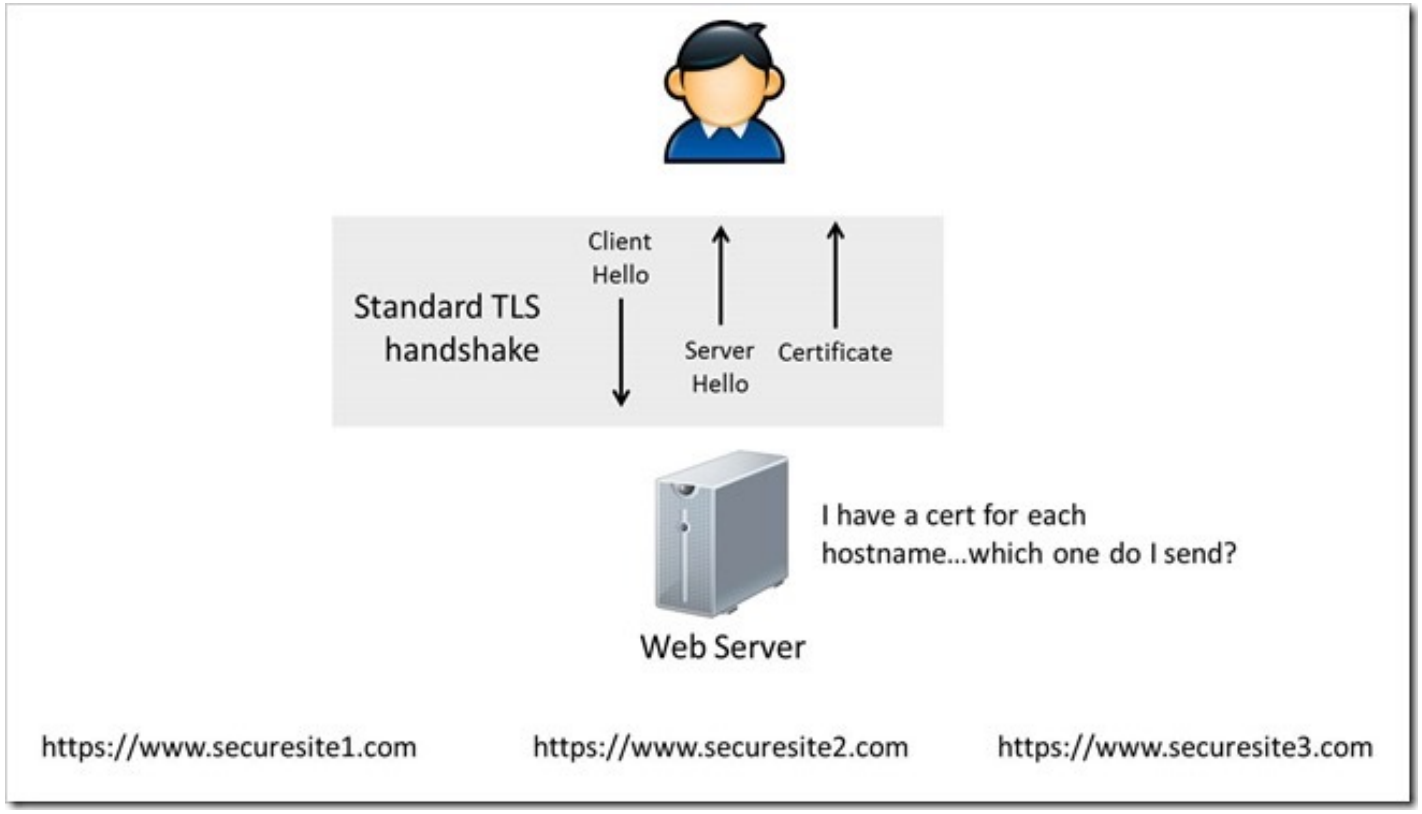
https://istlsfastyet.com/

HTTPS
Verkürzter Handshake



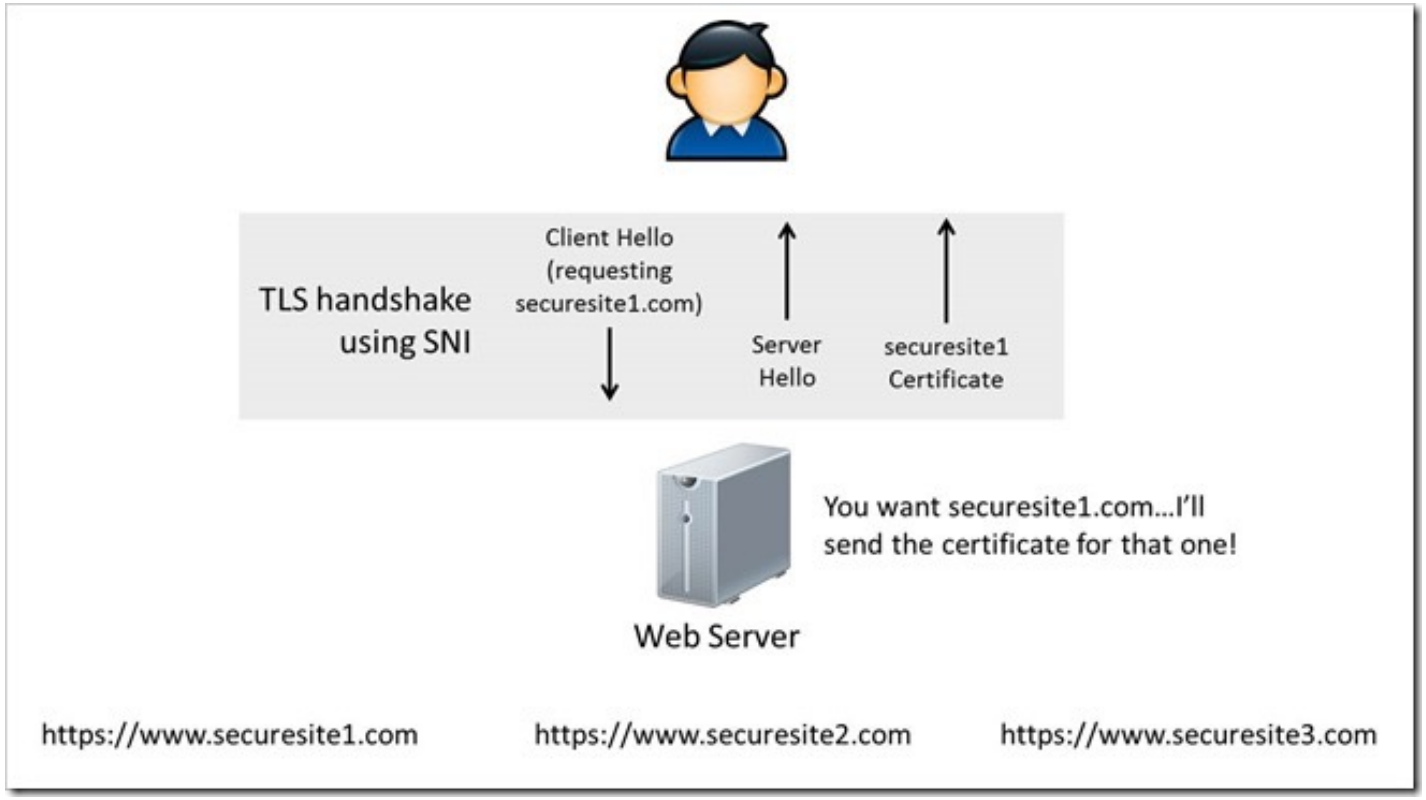
https://istlsfastyet.com/

TLS und Virtual Hosting
Server Name Indication



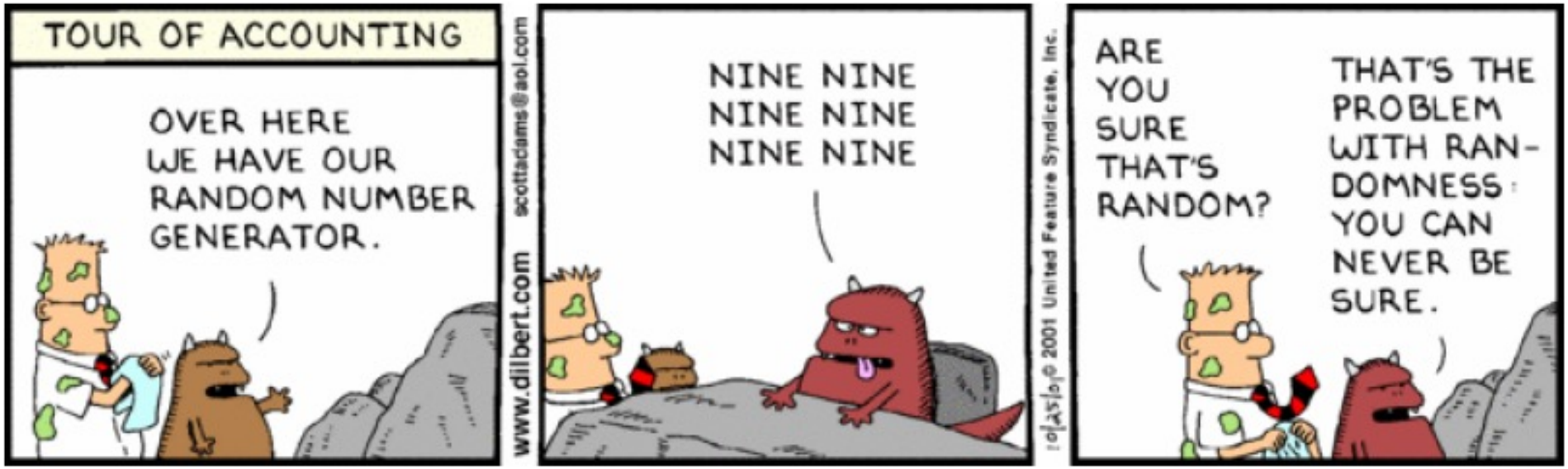
https://devcentral.f5.com/articles/ssl-profiles-part-7-server-name-indication

TLS und Virtual Hosting
Server Name Indication



https://devcentral.f5.com/articles/ssl-profiles-part-7-server-name-indication

TLS
Angriffspunkte?



http://dilbert.com/strip/2001-10-25

TLS
Angriffspunkte? Buffer Overflow

Problem:

- Nachlässige Programmierung
- Unsichere Programmiersprache (meist C)

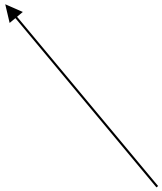
  ➔ Unzureichende Längenprüfung / Absicherung von Eingabedaten

Angriffstechnik:

- Durch Eingabedaten mit Überlänge (→ lokale Variablen, Parameter) werden Teile des Stacks überschrieben
- Überschreiben der echten Rücksprungadresse
- Platzieren von eigenem Assemblercode auf dem Stack oder einer gefälschten „Rücksprungadresse" mit Aufruf einer Bibliotheksprozedur (LoadLibrary, Shell, ..)!

TLS
Angriffspunkte? Buffer Overflow

```
cmd = lies_aus_netz();
do_something(cmd);
.....
int do_something(char* InputString) {
        char buffer[4];
        strcpy (buffer, InputString);
        ...
        return 0;
}
```



`strcpy` kopiert ohne Prüfung solange in den Speicher, bis NULL gelesen wird!!!

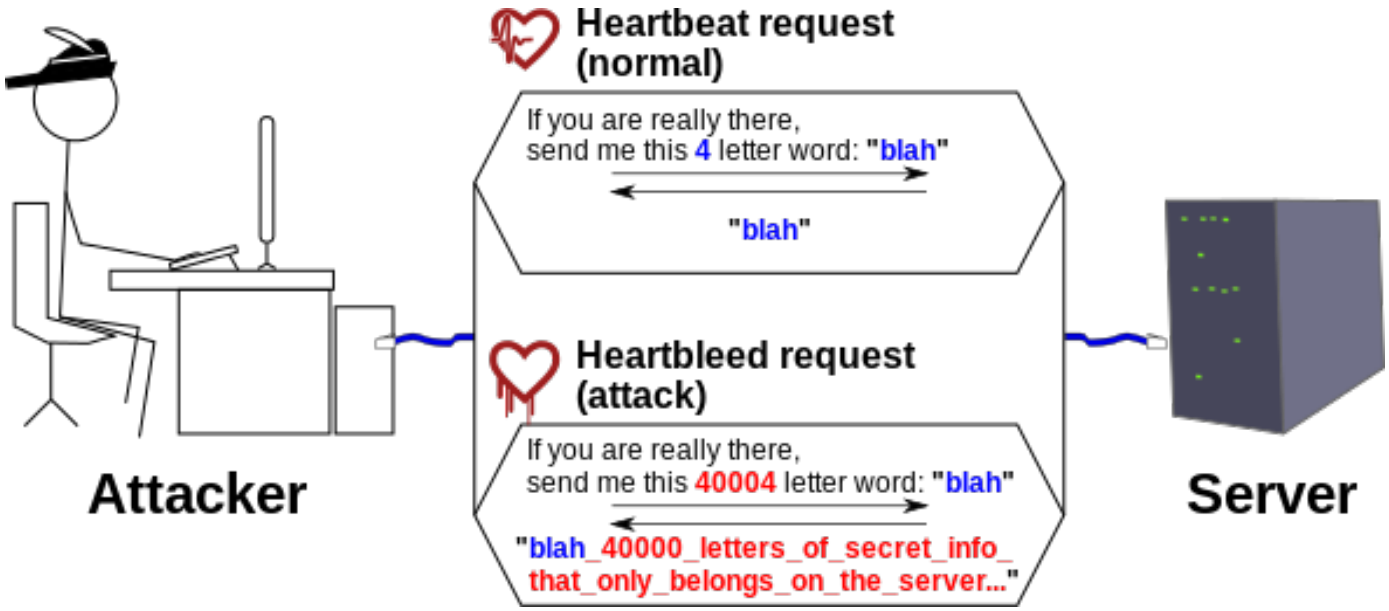
TLS/SSL
Angriffspunkte

Zufallszahlengenerator:

- Rechner haben nur einen Pseudozufallszahlengenerator
- Die Entropie der generierten Zahlen ist zu gering
- Von 2004 bis 2013 hatte eine Standardbibliothek zur Erzeugung von Zufallszahlen eine Backdoor für die NSA

Implementierungsfehler: Poodle,

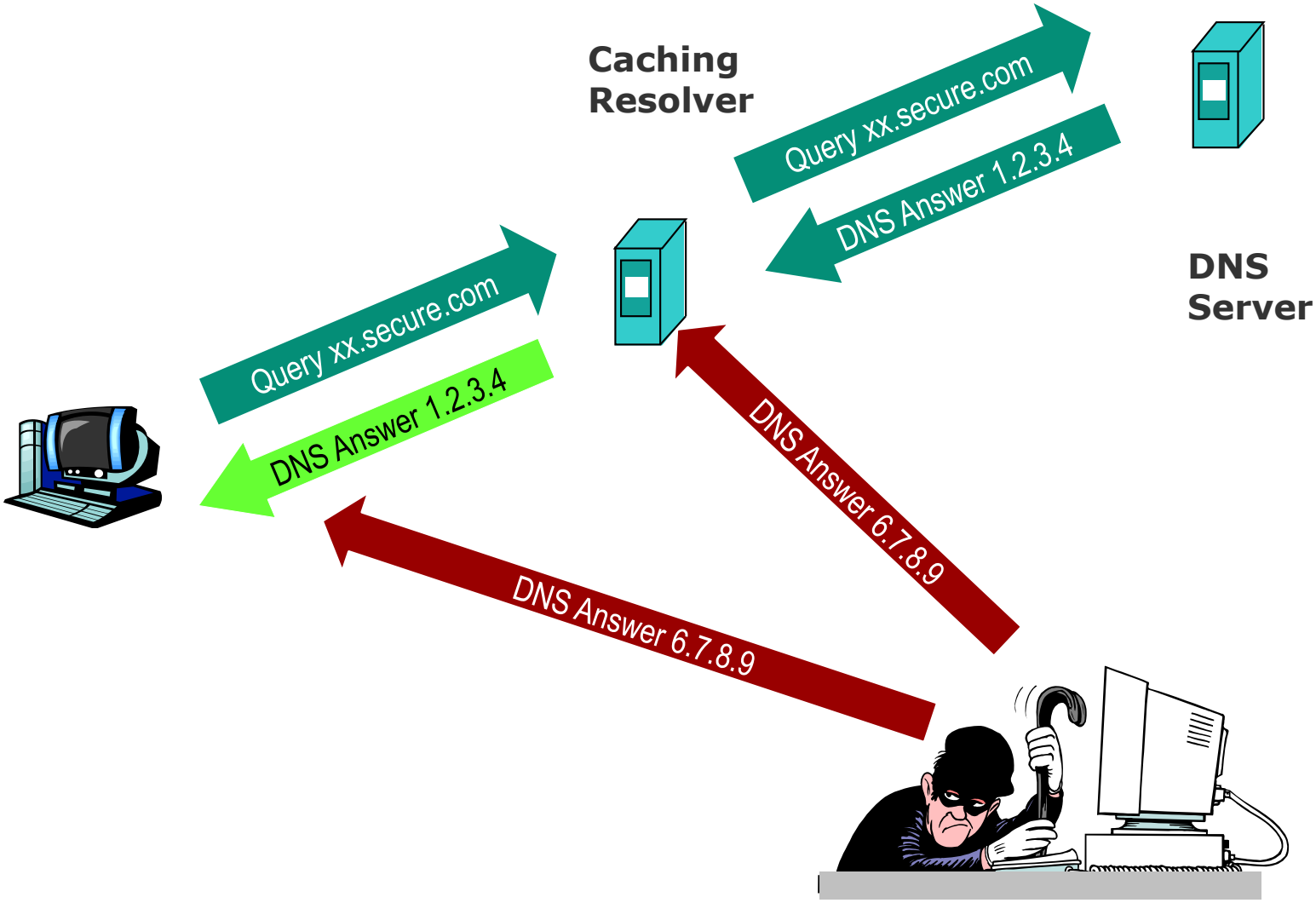
Heartbleed-Bug (OpenSSL):



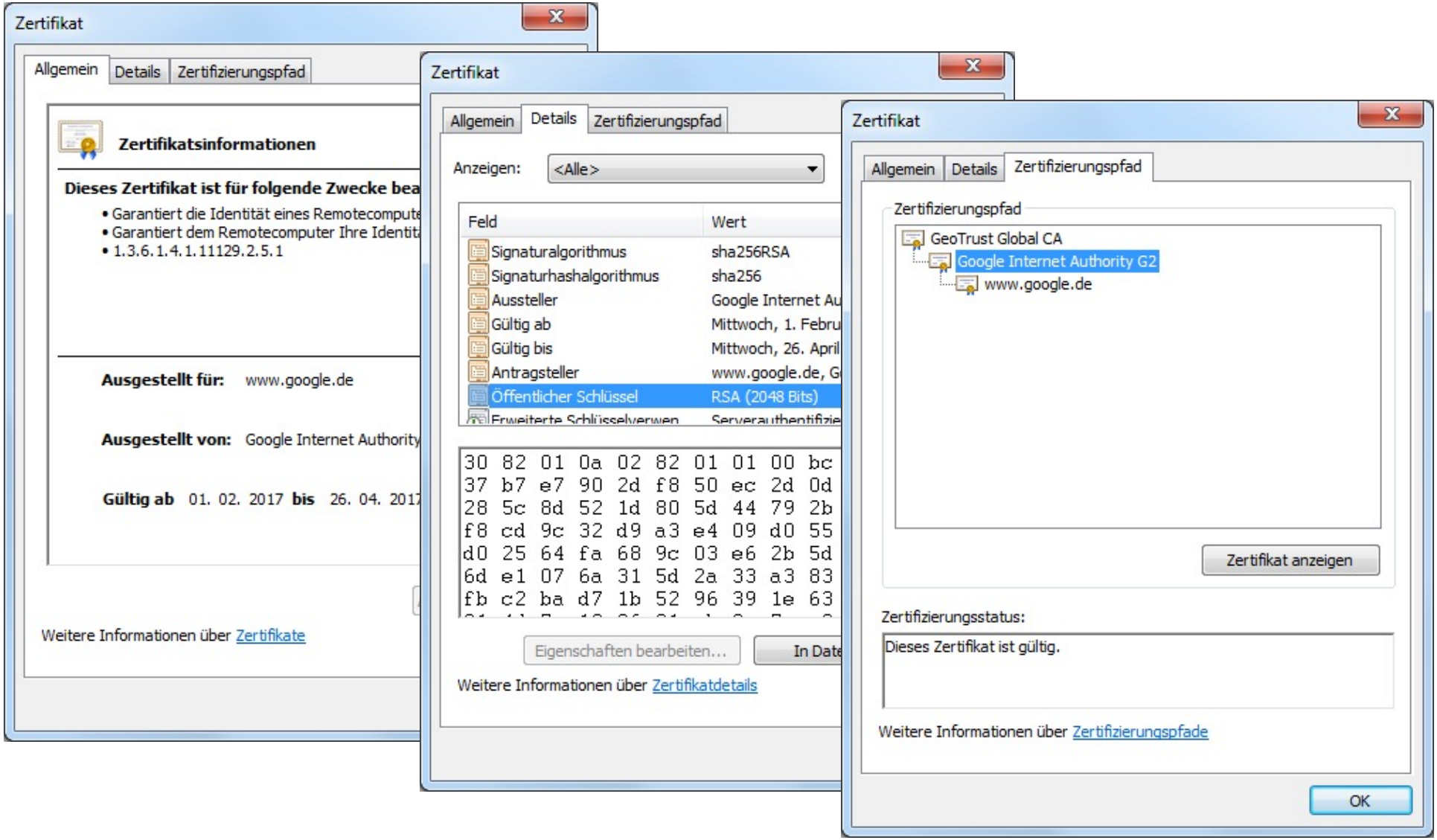
Länge des Payloads wurde vom Server nicht überprüft → Angreifer kann Speicher des Servers auslesen

https://upload.wikimedia.org/wikipedia/commons/thumb/c/cb/Heartbleed_bug_explained.svg/1024px-Heartbleed_bug_explained.svg.png

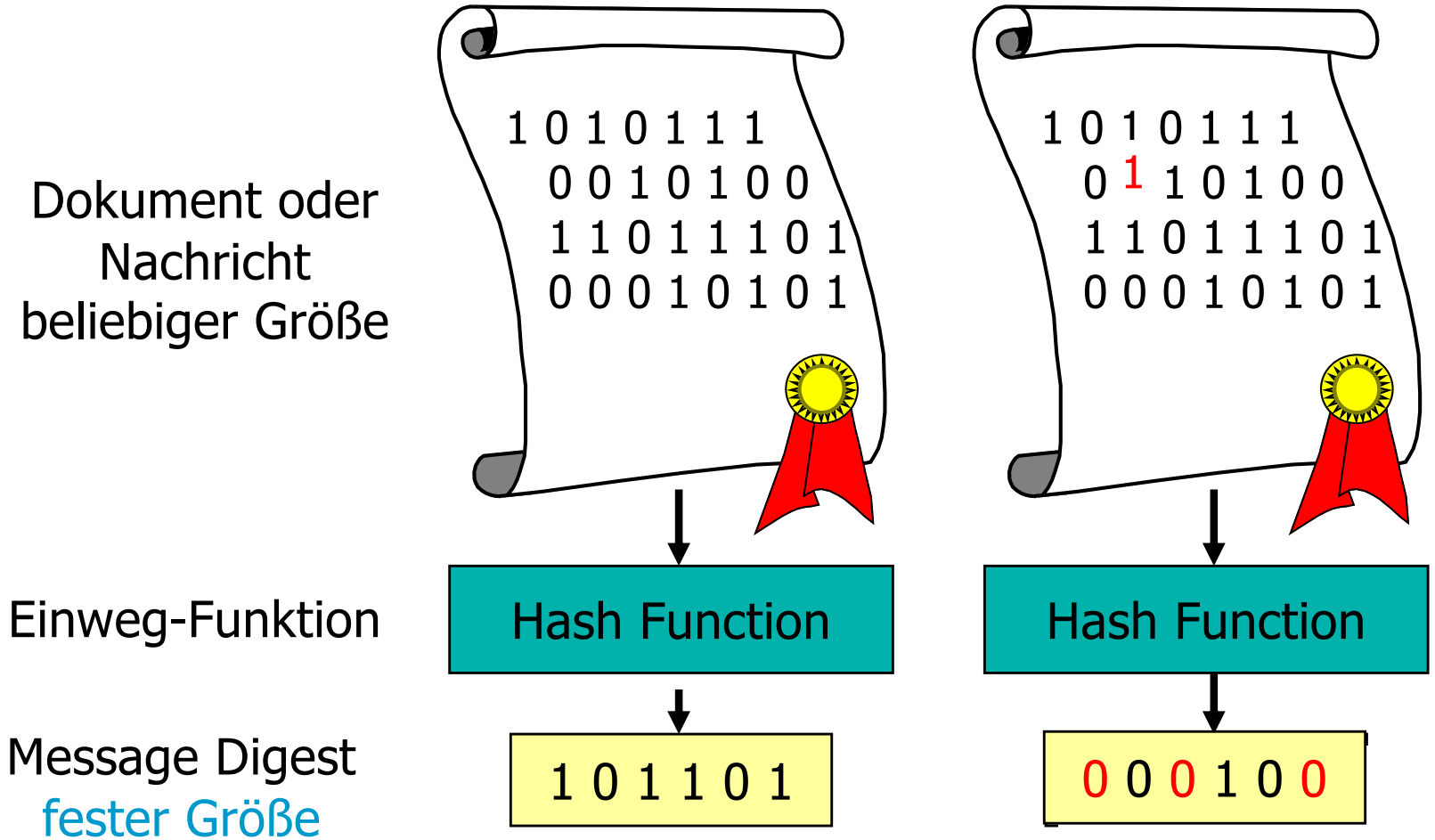
TLS/SSL
Angriffspunkte



HTTPS
Zertifikate

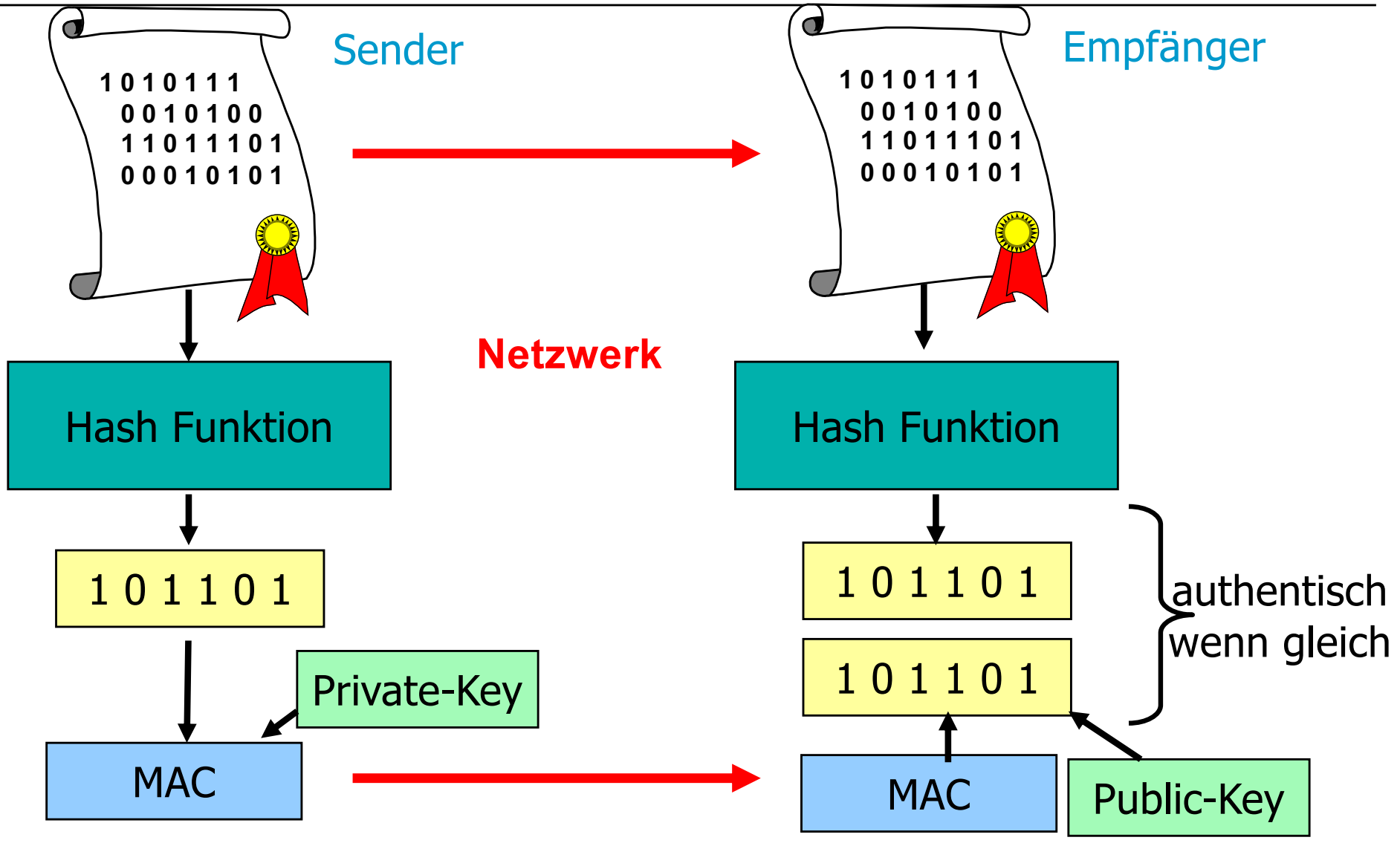


HTTPS
Signaturen

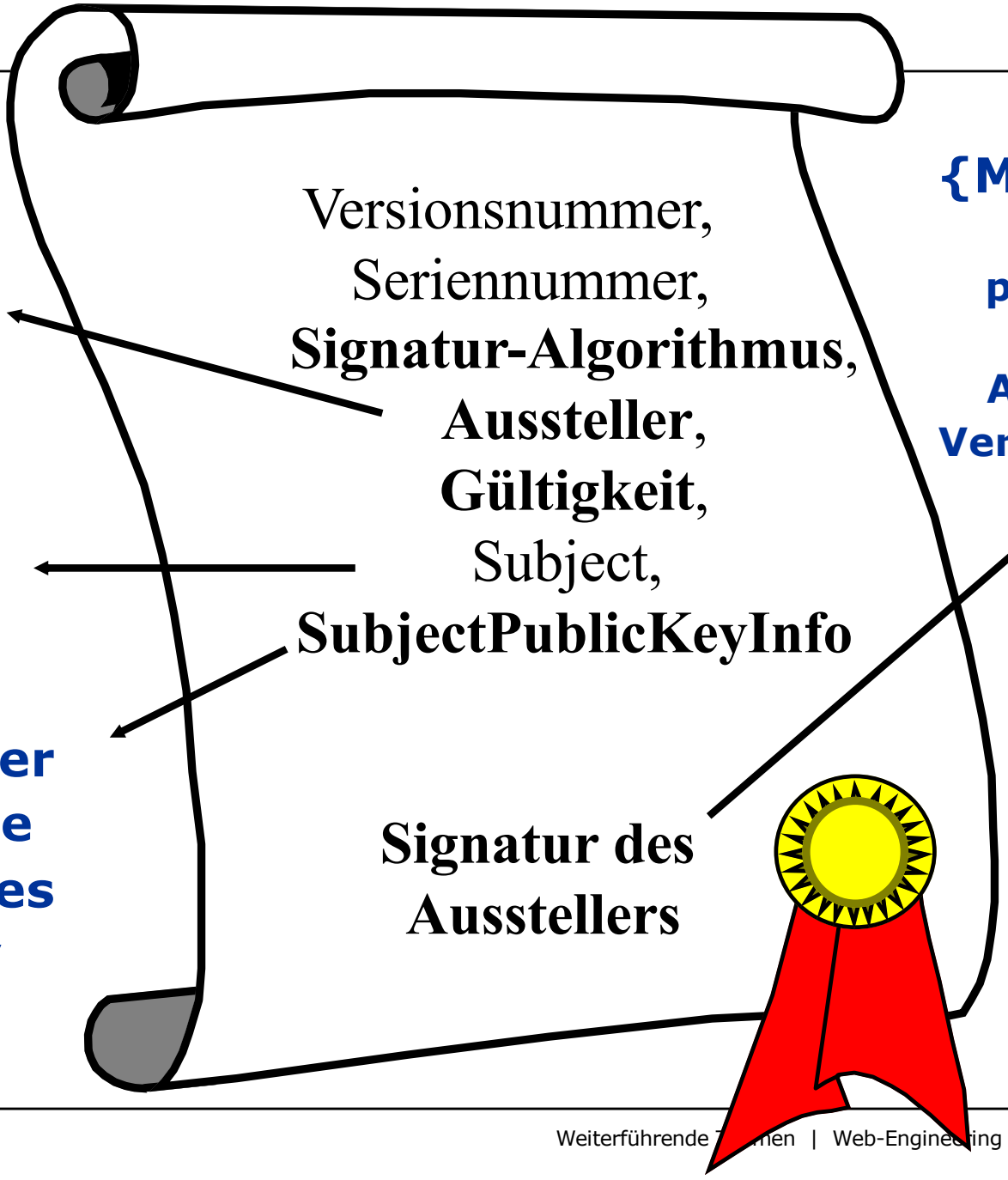


Die Änderung eines einzelnen Bits im Dokument sollte ungefähr 50% der Hash-bits verändern!

HTTPS
Einsatz von Signaturen



HTTPS
Zertifikate



Aussteller der die Identität validiert hat

Die zu prüfende Identität

Der Aussteller bestätigt die zuordnung des Public Key

{MD5 hash}
Mit dem private key des Ausstellers Verschlüsselter Hash

HTTPS
Certificate Authority (CA)

Vertrauen in Certificate Authorities

- CAs stellen u.a. für Webseitenbetreiber Zertifikate aus und signieren diese (damit benötigen diese auch ein Zertifikat)
- Vertraut der Client der CA, so vertraut er auch dem ausgestellten Zertifikat
- Browser und Betriebssysteme kommen mit einer Reihe voreingestellten „Trusted Root CAs“

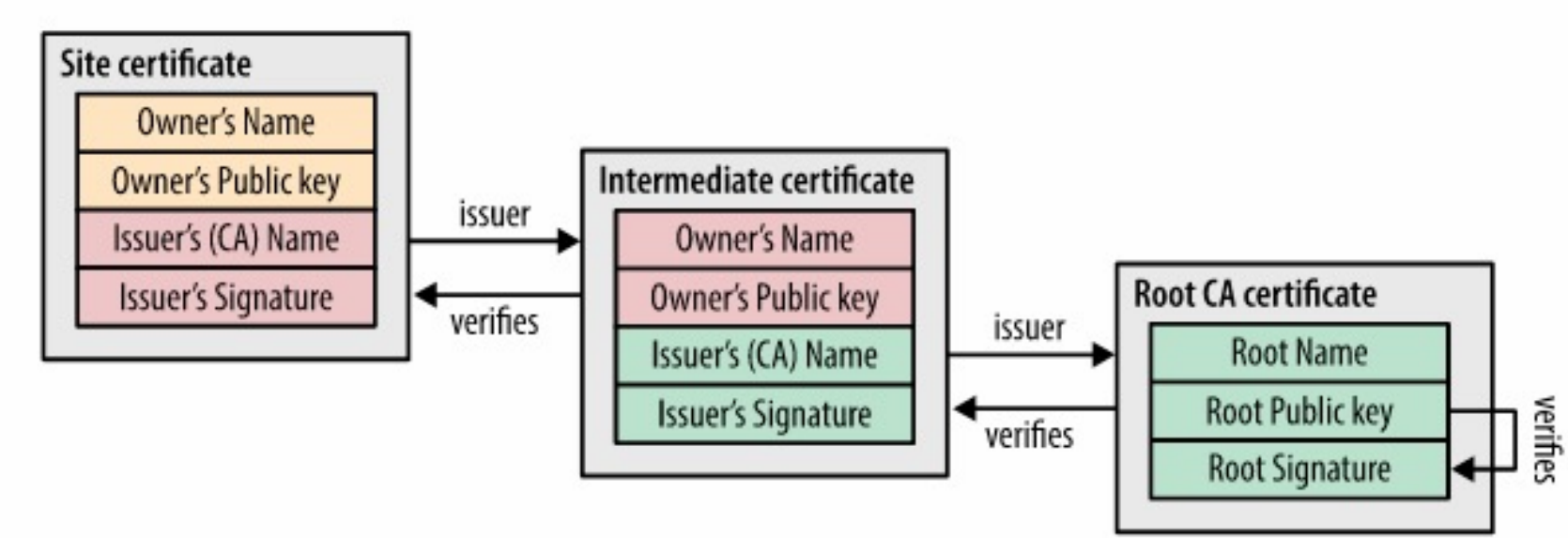
Trusted Root CA

- Kann Zertifikate für weitere CAs ausstellen
  - > Beispiel: GeoTrust CA -> Google CA
- Muss ein längeres Verfahren durchlaufen bis Browserhersteller oder Betriebssysteme die CA aufnehmen; **Selbst-Signiert!**

Trusted CA

- Kann beliebige Zertifikate ausstellen
- Muss i.d.R. ein längeres Verfahren durchlaufen bevor sie als CA akzeptiert werden

HTTPS
Vertrauensketten

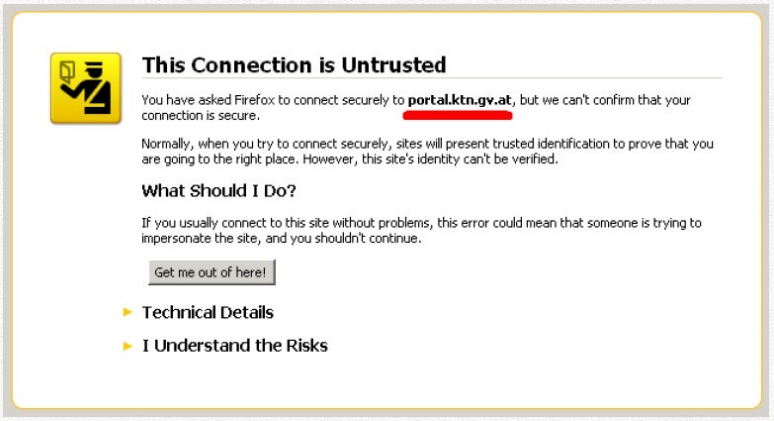


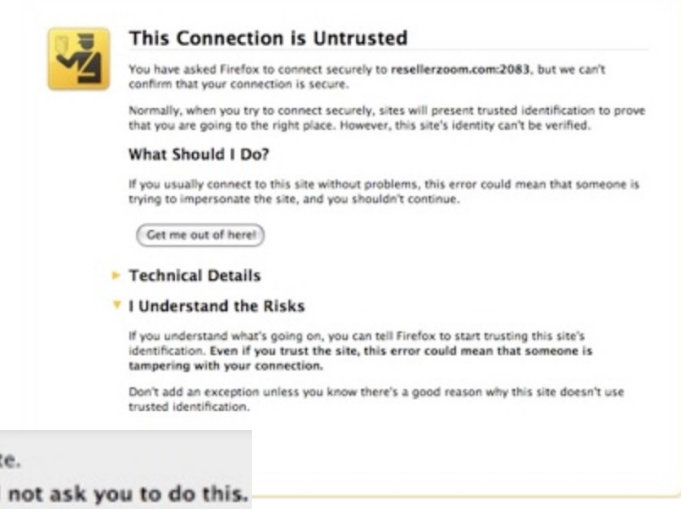
https://hpbn.co/transport-layer-security-tls/#tls-handshake

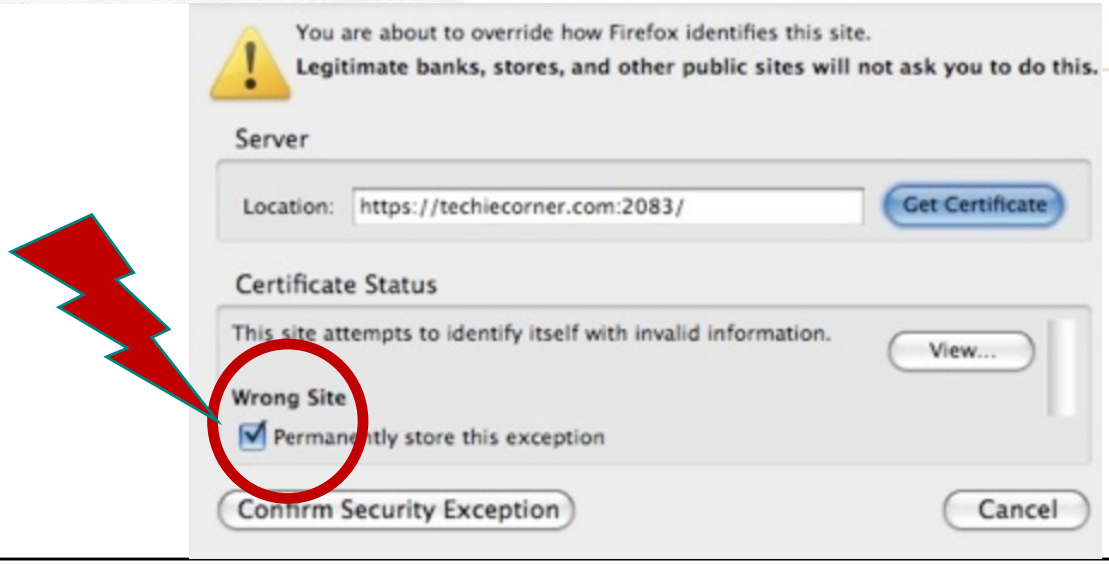
HTTPS
Certificate Authority (CA)

Browser speichern Trusted (Root) CAs

Falls die Web-Seite ein Zertifikat einer bisher nicht aufgelisteten CA liefert, so wird nachgefragt: Aufpassen!







HTTPS
Certificate Authority (CA)

Anzahl an Certificate Authorities

- Mozilla: 124 Trusted Root Zertifikate (~60 Organisationen)
- Microsoft: „nur" 19 Trusted Root Zertifikate in Windows 7
  - > Durch Updates erweitert Microsoft diese Liste kontinuierlich

EFF SSL Observatory

- Projekt, dass die Anzahl an CAs schätzt
- 1482 CA Zertifikate denen Windows oder Firefox vertraut
  - > Insgesamt 651 Organisationen

Ein einziger kompromittierter CA gefährdet das gesamte Web

- Zahlreiche Vorfälle in der Vergangenheit zeigen wie anfällig das Verfahren ist
  - > Bekannter Fall: DigiNotar 2011
    - – Angreifer kamen in Besitz des Private Keys der CA und stellten sich Zertifikate für über 500 Domains (u.a. google.com) aus, u.a. um iranische Bürger abzuhören
    - – In 2011 nutzten über 300.000 Seiten das gefälschte google-Zertifikat (99% im Iran)

Quellen & weitere Informationen: https://www.eff.org/observatory, https://www.eff.org/files/defconssliverse.pdf, https://heise.de/-1336603

Arten von X.509-Zertifikaten

- „Normales“ Zertifikat
  - > CA garantiert die Echtheit der Webseite (Organization Validated)
    - Prüfverfahren der CA des Antragsstellers der CA ggf. recht lose
  - > Kosten: ≥ 0 Euro (z.B. kostenlos über https://letsencrypt.org/)

- Extended-Validation Zertifikat
  - > CA garantiert die Echtheit und den Inhaber der Webseite
    - Feststellung der Identität und der Geschäftsadresse des Antragstellers
    - Sicherstellung, dass der Antragsteller ausschließlicher Eigentümer der Domain ist oder eine exklusive Nutzungsberechtigung hat
    - Sicherstellung, dass die antragstellenden Personen befugt sind und dass rechtlich bindende Dokumente von zeichnungsberechtigten Personen unterschrieben werden
  - > Das normale Zertifikat enthält nur die (technische) Aussage, ob die Webseite verschlüsselt ist
  - > Kosten: ≥ 200 Euro

HTTPS
Aber...

However, it all comes down to the human factor. An EV certificate means that the domain is owned by a registered legal entity. It does not mean that site is de facto and de jure trustworthy. In2016, Troy Hunt pointed out correctly, that the effectiveness of EV certificates depends on how people and organizations realize their value and act to protect them. Flash forward a few years later, and he's thinking the days of using EV certificates are gone.

Nevertheless, if EV certificates are to be successful it requires technical controls for managing their lifecycle that can be enforced without relying on the user.

https://www.keyfactor.com/blog/what-are-extended-validation-certificates-and-are-they-dead/

Google and Mozilla have decided to eliminate visual signals in their Chrome and Firefox desktop browsers of special digital certificates meant to assure users that they landed at a legitimate site, not a malicious copycat.

The certificates, dubbed "Extended Validation" (EV) certificates, were a subset of the usual certificates used to encrypt browser-to-server-and-back communications. Unlike run-of-the-mill certificates, EVs can be issued only by a select group of certificate authorities (CAs); to acquire one, a company must go through a complicated process that validates its legal identity as the site owner. They're also more expensive.

https://www.computerworld.com/article/3431667/chrome-firefox-to-expunge-extended-validation-cert-signals.html

HTTPS
Aber...

https://www.troyhunt.com/extended-validation-certificates-are-really-really-dead

> *Through our own research as well as a survey of prior academic work, the Chrome Security UX team has determined that the EV UI does not protect users as intended. Users do not appear to make secure choices (such as not entering password or credit card information) when the UI is altered or removed, as would be necessary for EV UI to provide meaningful protection.*

> *The effectiveness of EV has been called into question numerous times over the last few years, there are serious doubts whether users notice the absence of positive security indicators and proof of concepts have been pitting EV against domains for phishing.*

> *More recently, it has been shown that EV certificates with colliding entity names can be generated by choosing a different jurisdiction. 18 months have passed since then and no changes that address this problem have been identified.*

How to Get a VeriSign Certificate

- Pay VeriSign ($300)
- Get DBA license from city hall ($20)
  - No on-line check for name conflicts; can I do business as Microsoft?
- Letterhead from company (free)
- Notarize document (need driver's license) (free)
- Easy to get fradulent certificate
  - Maybe hard to avoid being prosecuted afterwards...
- But this is just VeriSign's policy
  - **many** other CAs...

http://www0.cs.ucl.ac.uk/staff/B.Karp/gz03/f2016/lectures

Public Key Infrastructure
Exkurs

In mid-March 2001, VeriSign, Inc., advised Microsoft that on January 29 and 30, 2001, it issued two VeriSign Class 3 code-signing digital certificates to an individual who fraudulently claimed to be a Microsoft employee. The common name assigned to both certificates is "Microsoft Corporation".

https://docs.microsoft.com/en-us/security-updates/securitybulletins/2001/ms01-017

Public Key Infrastructure
Exkurs

Stuxnet (enthielt Zertifikate für JMicron & Realtek)

Verwendet gestohlene Zertifikate

Entdeckt von ESET

Public Key Infrastructure
Motivation

Der Einsatz von Zertifikaten erfordert

- Das Wissen um die Prozesse zur Erstellung von Zertifikation
  - > Wie wird die Identität sicher gestellt
  - > Wird alles von einer Organisation durchgeführt
- Die Pflege von Zertifikatsperrlisten (certificate revocation lists)
  - > Zertifikate können ihren Zweck verlieren, z.B. bei Verlust des Private Keys
  - > Listen mit zurückgezogenen, abgelaufenen und für ungültig erklärten Zertifikaten

Organisation, die einen PKI-Dienst anbietet: **Trust-Center**

Gesetzliche Vorschriften

- Signaturgesetz von 1997
- EU-Richtlinie
- Signaturgesetz (SigG) von 2000
- Signaturverordnung
- Gesetz zur Anpassung der Formvorschriften (2001)

Public Key Infrastructure (PKI)
Bestandteile

Registrierungsstelle (Registration Authority, RA):

- Organisation, bei der Personen, Maschinen oder auch untergeordnete Zertifizierungsstellen Zertifikate beantragen können
- Diese prüft die Richtigkeit der Daten im gewünschten Zertifikat und genehmigt den Zertifikatsantrag der dann durch die Zertifizierungsstelle signiert wird

Zertifizierungsstelle (Certificate Authority, CA):

- Vertrauenswürdige Organisation, die die Signatur von Zertifikatsanträgen gemäß bekannter Regeln übernimmt

Verzeichnisdienst:

- ein durchsuchbares Verzeichnis, welches ausgestellte Zertifikate enthält

Validierungsdienst:

- Ein Dienst, der die Überprüfung von Zertifikaten in Echtzeit ermöglicht

HTTP
Strict Transport Security (**HSTS**)

Problem: Ausschalten oder Downgrading der TLS-Verschlüsselung mittels redirect u.a. möglich

- POODLE-Attacke: Padding Oracle On Downgraded Legacy Encryption
  - > 2014 konnte mittels eines Downgrade Dance die Verbindung auf SSL Version 3.0 (pre TLS) reduziert werden
  - > Man-in-the-Middle-Attacke um darauf hin Klartext zur Kommunikation mit dem echten Server zu erhalten
- Durch Verwenden des HSTS-Headers wird dem Webbrowser mitgeteilt, dass eine Seite nur über https und den angegebenen Cipher Suites erreichbar ist
- Sobald man einmal mit der Webseite verbunden war merkt sich der Browser, die Verbindungseigenschaften

Ergänzung durch Pinning

- Anschaulich wird der öffentliche Schlüssel des Browser-Zertifikats „festgenagelt“, durch einen weiteren Hash gesichert
- Dann kann niemand versuchen, die Identität des Servers mit einem gefälschten Zertifikat anzunehmen

HTTP
Strict Transport Security (**HSTS**)

```
const hsts = require('hsts')

app.use(hsts({
  maxAge: 15552000  // 180 days in seconds
}))
// Strict-Transport-Security: max-age: 15552000; includeSubDomains
```

https://www.npmjs.com/package/hsts

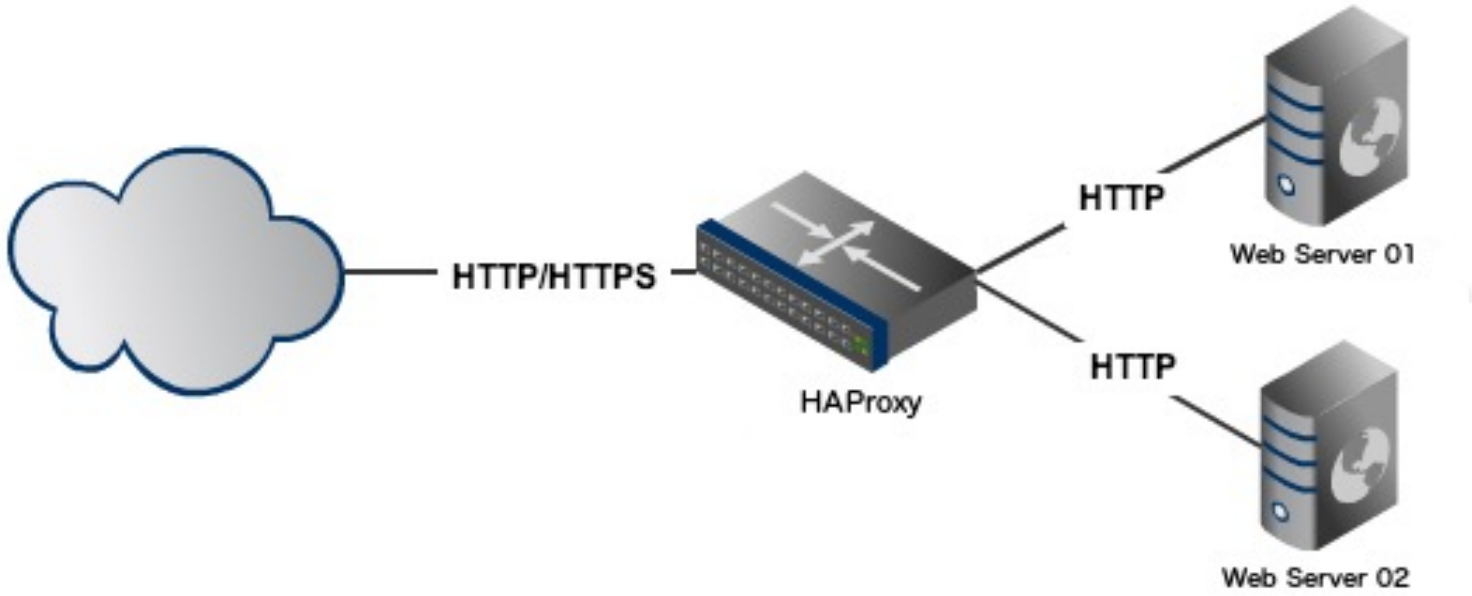
Dem Browser wird über den Strict-Transport-Security -Header mitgeteilt, dass er für die nächsten 180 Tage https benutzen soll

```
// Make sure you run "npm install helmet" to get the Helmet package.
const helmet = require('helmet')

// Sets "Strict-Transport-Security: max-age=5184000; includeSubDomains".
const sixtyDaysInSeconds = 5184000
app.use(helmet.hsts({
  maxAge: sixtyDaysInSeconds
}))
```

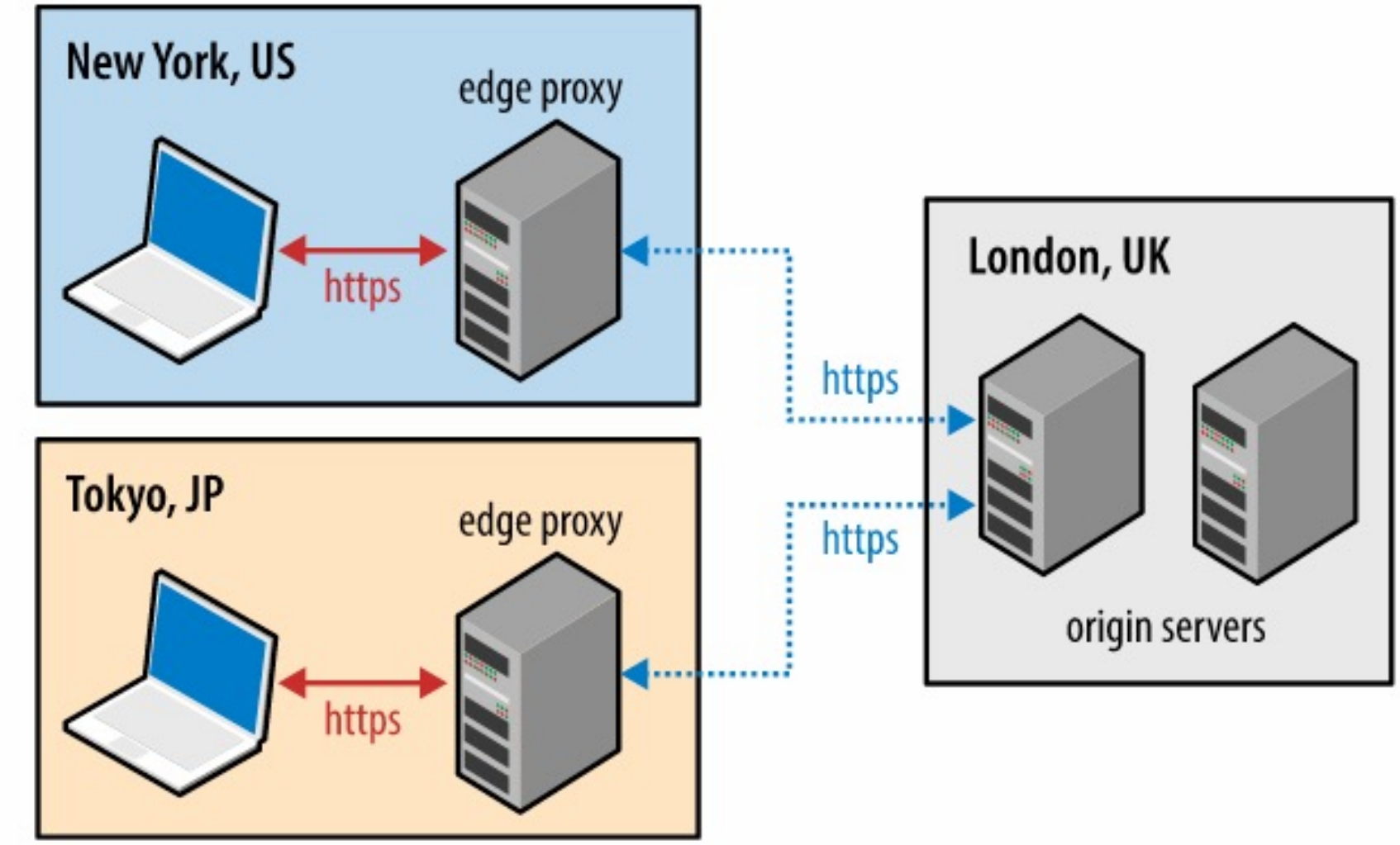
“Helmet” ist eine Ansammlung von neun kleineren Middleware-Angebote, über die sicherheitsrelevante HTTP-Header (u.a. hsts) festgelegt werden.

HTTPS
TLS-Terminierung beim Revers Proxy (NGINX)



https://www.howtoforge.com/tutorial/ubuntu-load-balancer-haproxy/

HTTPS
TLS Early Terminierung – SSL Visibility Appliance



https://hpbn.co/transport-layer-security-tls/#tls-handshake