IAIK TU Graz

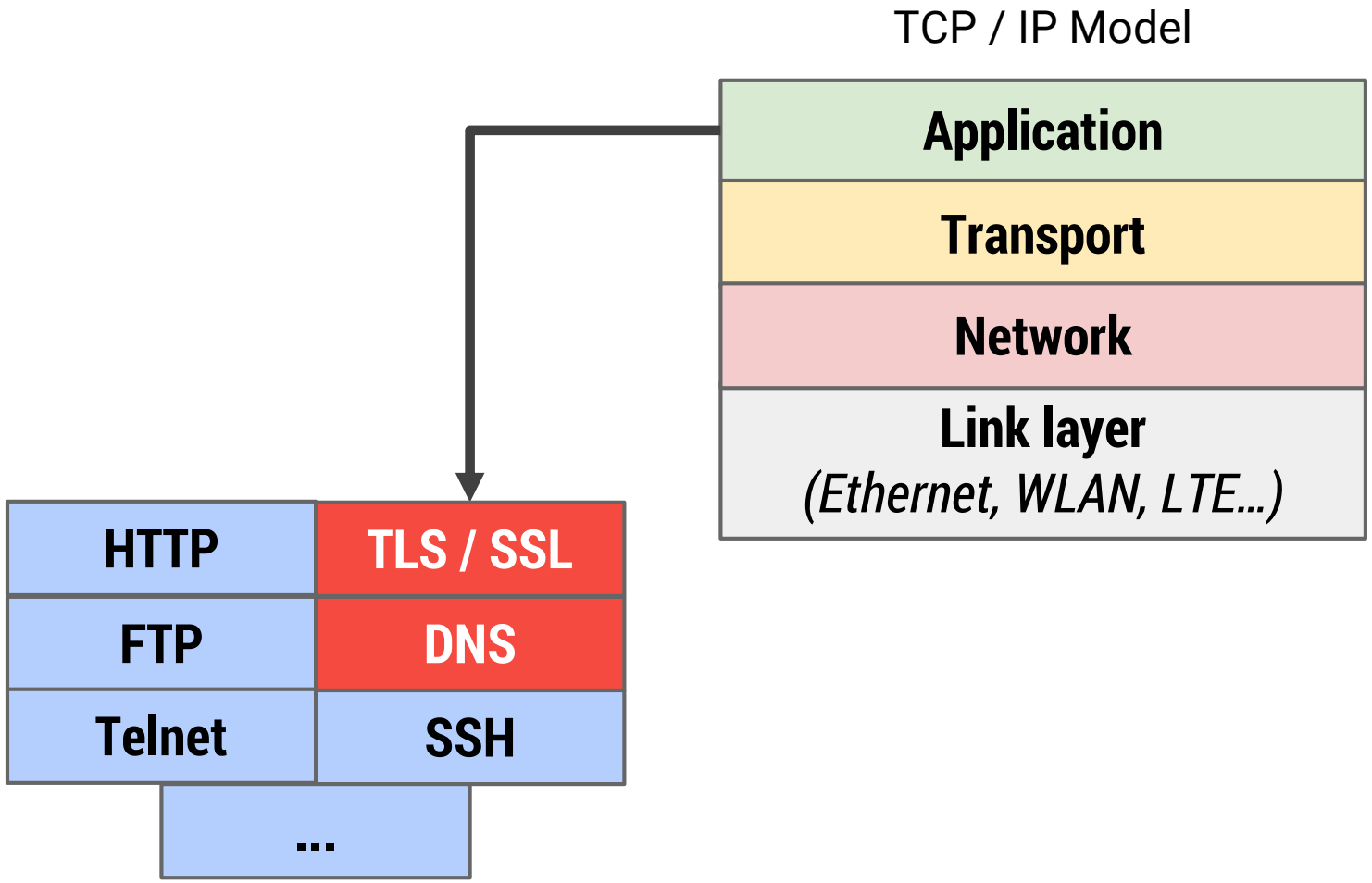
TLS Attacks & DNS Security

Information Security 2019

Johannes Feichtner
johannes.feichtner@iaik.tugraz.at

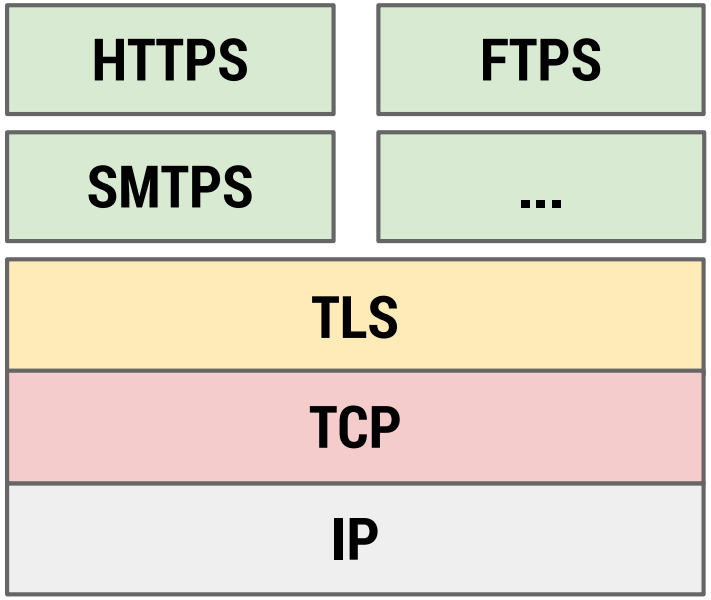
Outline

- Browser Issues
  - SSLStrip
  - MITM Attack revisited
- PKI Attacks
  - Weaknesses
  - FLAME
- Implementation Attacks
- Protocol Attacks
- DNS Security



Review: TLS Services

All applications running TLS are provided with three essential services



Authentication
Verify identity of client and server

Data Integrity
Detect message tampering and forgery, e.g. malicious Man-in-the-middle

Encryption
Ensure privacy of exchanged communication

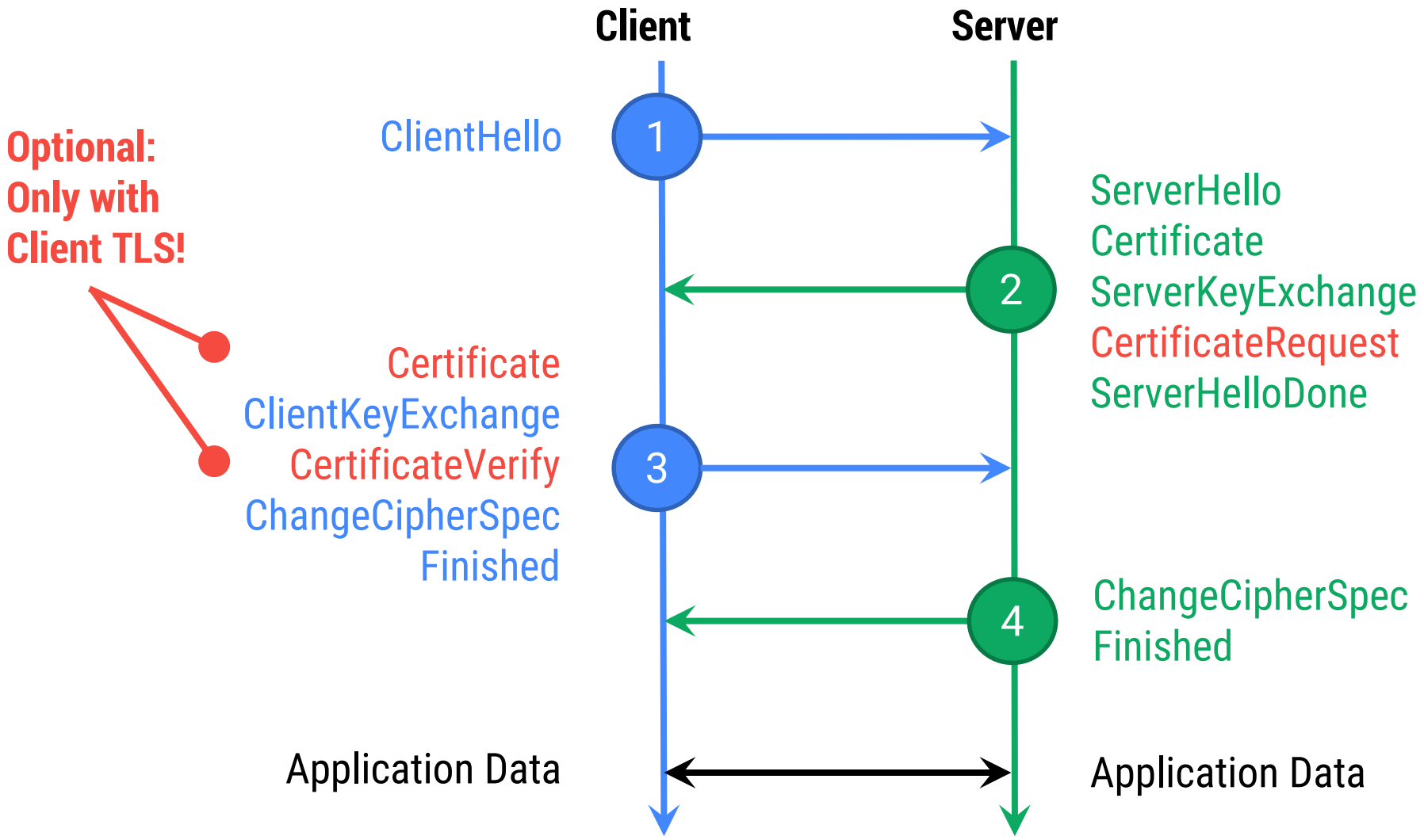
Note: Technically, not all services are required to be used
→ Can raise risk for security issues!

Review: TLS Handshake

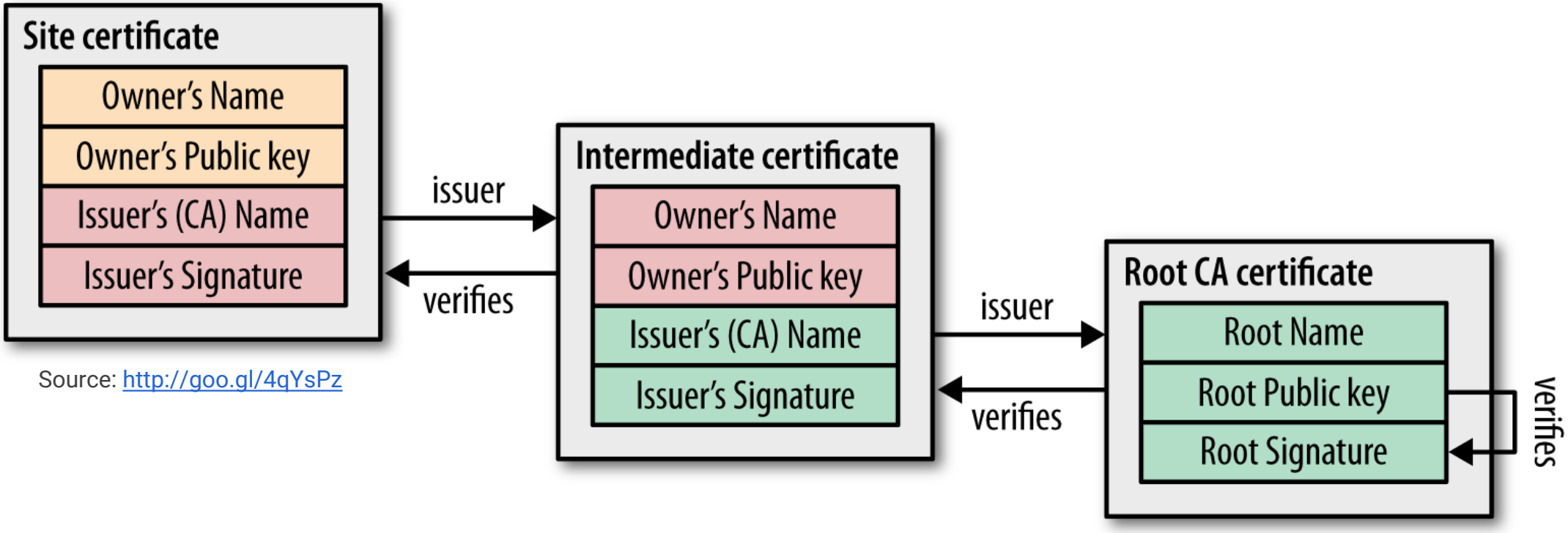
RFC 5246

= Establish parameters for cryptographically secure data channel

Full handshake scenario!



Review: Certificates



Source: http://goo.gl/4qYsPz

- Certificate Authority (CA) = Third party, trusted by both the subject (*owner*) of the certificate and the party (*site*) relying upon the certificate
- Browsers ship with set of > 130 trust stores (root CAs)

Browser Issues

Overview

Focus: Relationship between TLS and HTTP

Problem?

- Attacker wants to access encrypted data
- Browsers also have to deal with legacy websites
  → Enforcing max. security level would „break“ connectivity to many sites

Attack Vectors

- SSLStrip
- MITM Attack

…and somehow related: Cookie Stealing due to absent „Secure“ flag…

Review: ARP Poisoning

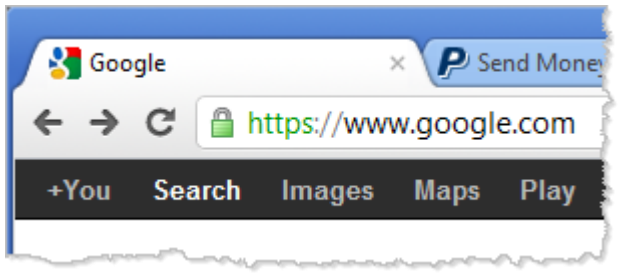
How?

a) Join WLAN,
start ARP Poisoning

b) Create own AP
   - E.g. with smartphone…

Attacker

- Sniff data
- Manipulate data
- Attack HTTPS connections



Client

http://www.apple.com
http://www.microsoft.com
https://www.google.com

SSLStrip

Or more accurately: „HTTPS Stripping“

Problem

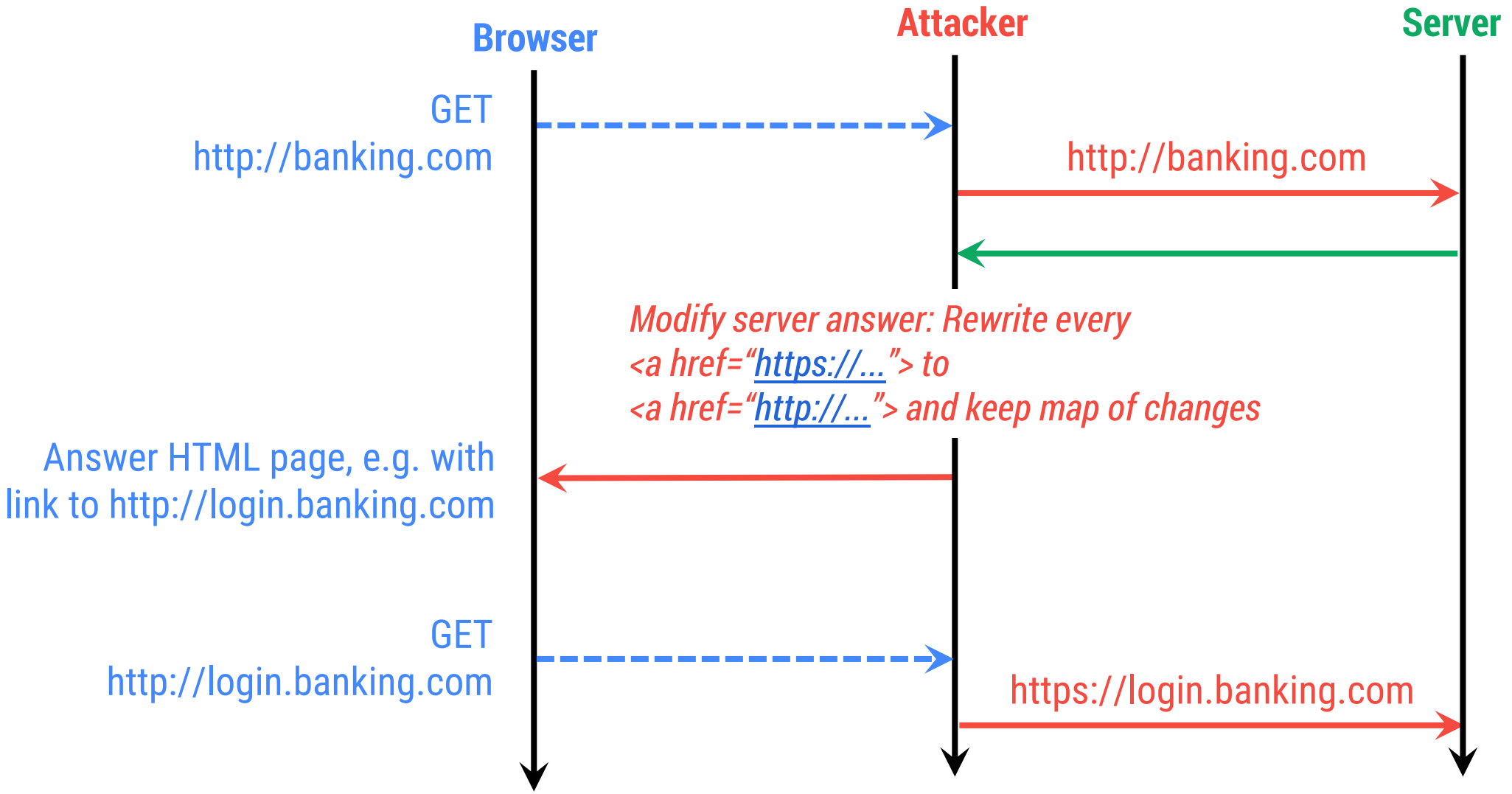
- Who types https:// when calling URLs?
  - Typically, scheme prepended by browser, by clicking on links, or through redirects
- If no prefix specified, browsers **try http:// first**

Idea

1. Perform MITM attack (ARP Poisoning) on unencrypted HTTP transmission
2. Rewrite content to replace https:// links with http:// equivalents
   - Prevent victim from accessing encrypted resources
3. Proxy HTTP requests to genuine HTTPS destination

SSLStrip

Protection? HSTS!



MITM Attack

…by faking server certificates

Problem

- Users often accept invalid or self-signed certificates anyway

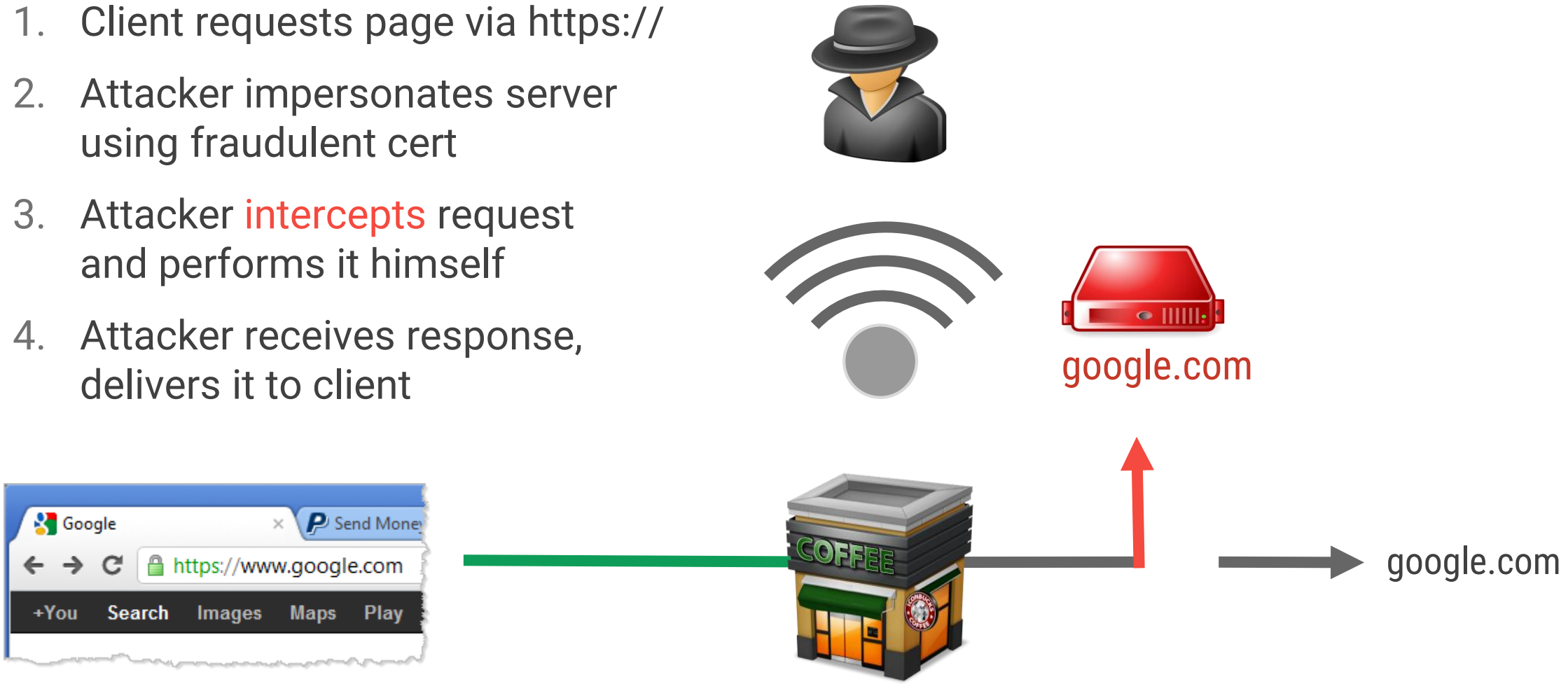
- We have ~130 certificate authorities (CA) in our browsers‘ trust stores
  - They are not equally rigid when issuing certificates

  → „Rogue certificate“ could be obtained and misused

- Exploit validation flaws - especially with mobile apps
  - Can overwrite certificate validation routines
  - Many apps silently (without warning) accept invalid certificates

MITM Attack

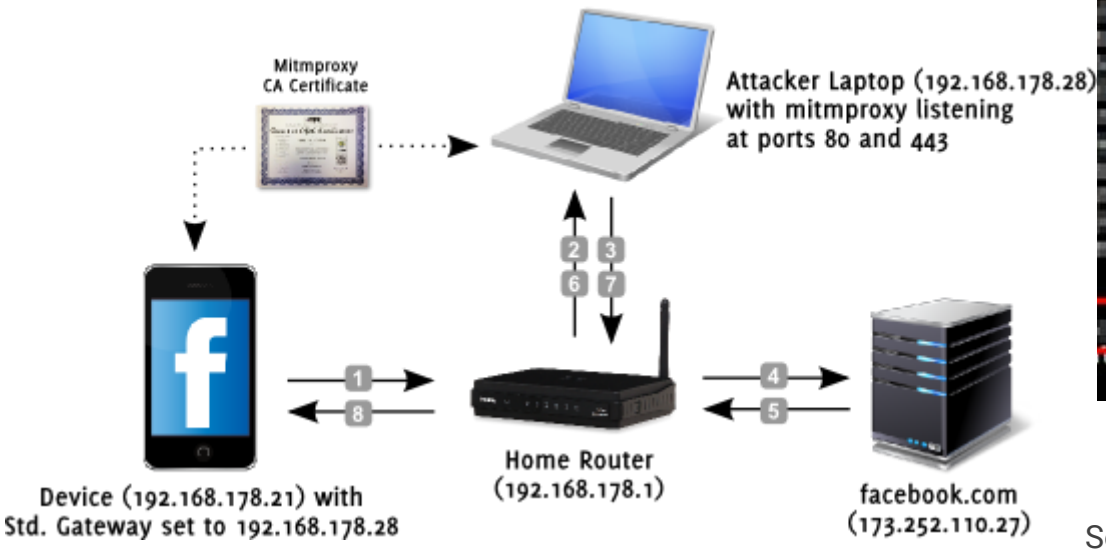
1. Client requests page via https://
2. Attacker impersonates server using fraudulent cert
3. Attacker intercepts request and performs it himself
4. Attacker receives response, delivers it to client



MITM Attack

Tools

- sslsplit
- mitmproxy
- Fiddler
- Burp Suite



Source: http://goo.gl/EjihVg

```
root@pbox: /tmp/sslsplit/logdir
root@pbox: ~/sslsplit-0.4.7 129x11
===> Original server certificate:
Subject DN: /C=US/ST=California/L=Palo Alto/O=Facebook, Inc./CN=*.facebook.com
Common Names: *.facebook.com/*.facebook.com/facebook.com
Fingerprint: f5:6b:f2:44:63:b0:bd:61:36:c5:e8:72:34:6b:32:04:28:ff:4d:7c
Certificate cache: HIT
===> Forged server certificate:
Subject DN: /C=US/ST=California/L=Palo Alto/O=Facebook, Inc./CN=*.facebook.com
Common Names: *.facebook.com/*.facebook.com/facebook.com
Fingerprint: 9c:01:9c:40:13:5a:c4:c7:6c:5a:49:95:45:b5:f4:e0:d0:9c:e2:84
ssl [192.168.178.20]:39363 [31.13.81.33]:443 sni:www.facebook.com crt:*.facebook.com/*.facebook.com/facebook.com origcrt:*.facebo
ok.com/*.facebook.com/facebook.com
```

Forged certificate with a different fingerprint

```
root@pbox: /tmp/sslsplit/logdir 129x29
POST /login.php?login_attempt=1 HTTP/1.1
Host: www.facebook.com
Connection: keep-alive
Content-Length: 158
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Origin: https://www.facebook.com
User-Agent: Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/27.0.1453.93 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: https://www.facebook.com/index.php?stype=lo&jlou=
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Cookie:
%40gmail.com&pass=                    &persistent=1&default_persistent=1&timezone=-120&lgnrnd=
```

Source: https://goo.gl/iKnd7J

IAIK TU Graz

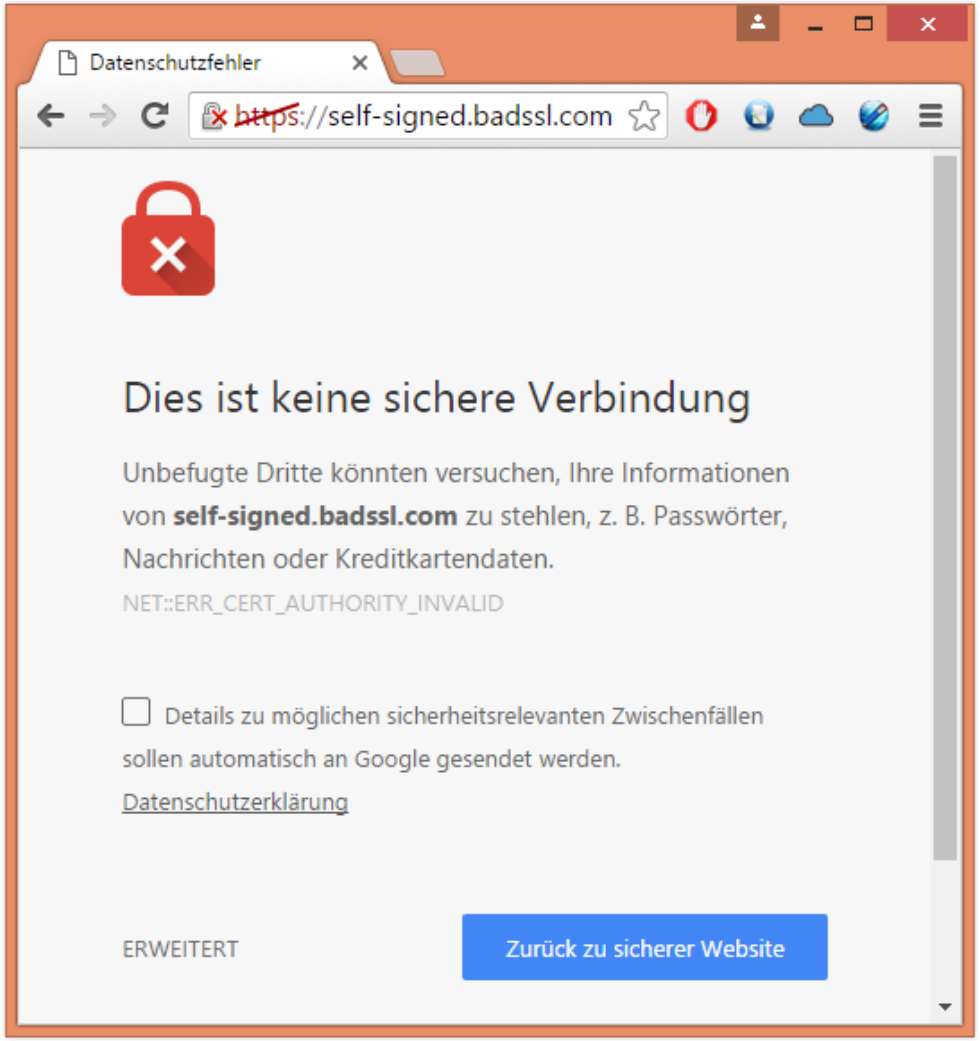
Certificate Warnings

What do you do if you receive an alert?

Many users proceed anyway!

→ 33% of Firefox users See: https://goo.gl/6gcir5

→ 56% of Chrome users See: http://goo.gl/S2oW8y



Why so many invalid certs?

- Misconfiguration of server
  - E.g. cert does not match domain name
- Certificate issues
  - User called domain name (*www.domain.at*) but cert only valid for domain.at
  - Validity expired

PKI Attacks

Overview

Public Key Infrastructure (PKI)

- Goal: Enable secure communication of parties that have never met
- Principles: *Identity, Authority, Trust*

But what is Trust (not)?

- Basically, just says that certificate can be validated by a CA in our trust store
- Trust can be inherited by intermediate CAs

→ Certificate Authorities (CAs) decide what is trustworthy!

Note: 46 countries with valid CAs See: https://goo.gl/VAYROa

USA, South Africa, England, Belgium, Japan, Germany, Netherlands, Israel, Saudi Arabia, Iceland, Russia, Macedonia … → *do you trust them?*

PKI Weaknesses

- Permission of domain owners not required for certificate issuance
  - *Any* CA can issue certificate for *any* domain without permission
- Weak domain validation
  - „Domain-validated“ certificates issued based on whether you control a domain, e.g. confirm mail receipt to *webmaster@domain.com* or place file within directory
- Revocation does not work
  - Theory: Browsers check against blacklists (CRL / OCSP) whether cert is blacklisted
  - Practice: If check takes too long → „*soft-fail*“ without error message
- Trust is not agile
  - We either trust a CA or not → nothing in between, e.g., grading

Root Key Compromise

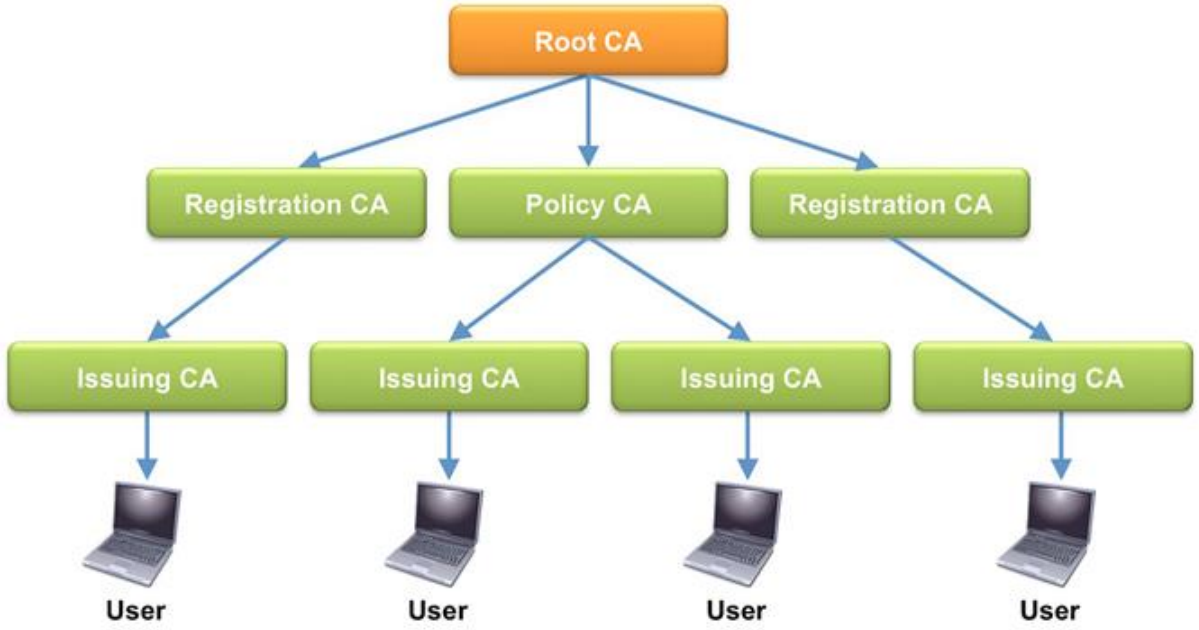
Best attack for PKI attack: Get root key

Source: http://goo.gl/QAYdNL

How?

- Steal them See: https://goo.gl/eCehMN
  - Harder because keys often enclosed in Hardware Security Module (HSM)
- Governmental agency
  → simply request them from CAs
- Break root (or intermediate) certs
  - Until 2014: Still (weak) 1024-bit RSA certificates in Firefox

See: https://goo.gl/T7cN66



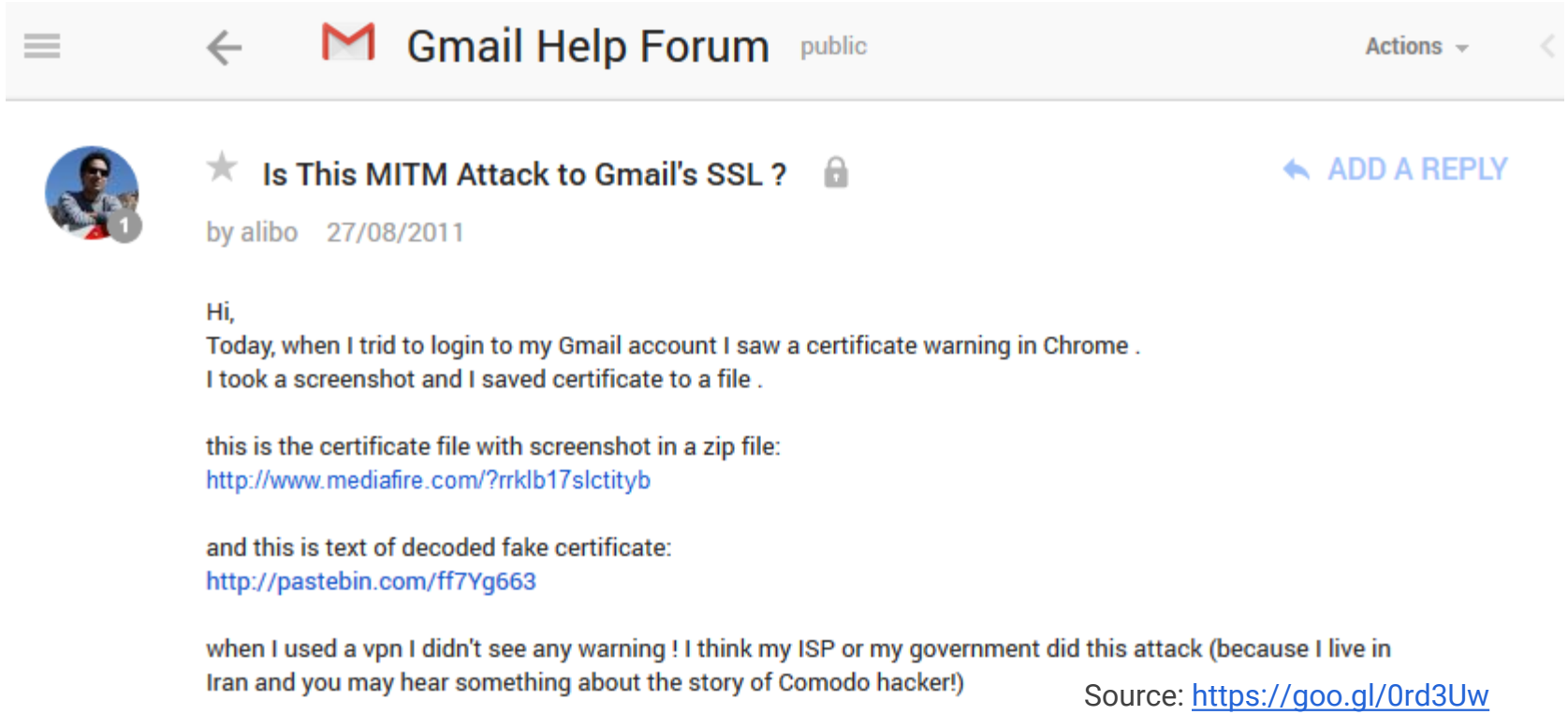
Source: https://goo.gl/SRuWKQ

CA Breach – DigiNotar

DigiNotar
Internet Trust Services

The first CA to be completely compromised…

- Public discovery on 27.08.2011 – obviously hacked since July
- Iranian user had problems accessing his mail account



Gmail Help Forum public Actions

Is This MITM Attack to Gmail's SSL ?

ADD A REPLY

by alibo 27/08/2011

Hi,
Today, when I trid to login to my Gmail account I saw a certificate warning in Chrome .
I took a screenshot and I saved certificate to a file .

this is the certificate file with screenshot in a zip file:
http://www.mediafire.com/?rrklb17slctityb

and this is text of decoded fake certificate:
http://pastebin.com/ff7Yg663

when I used a vpn I didn't see any warning ! I think my ISP or my government did this attack (because I live in Iran and you may hear something about the story of Comodo hacker!)

Source: https://goo.gl/0rd3Uw

IAIK TU Graz

CA Breach – DigiNotar

Some days later, it become public that the problem is (a lot) bigger…

The most critical servers contain malicious software that can normally be detected by anti-virus software.... We have strong indications that the CA-servers, although physically very securely placed in a tempest proof environment, were accessible over the network from the management LAN.

- The network has been severely breached. All CA servers were members of one Windows domain, which made it possible to access them all using one obtained user/password combination. The password was not very strong and could easily be brute-forced.

- The software installed on the public web servers was outdated and not patched.

- No antivirus protection was present on the investigated servers.

- An intrusion prevention system is operational. It is not clear at the moment why it didn't block some of the outside web server attacks. No secure central network logging is in place.

CA Breach – DigiNotar

Attacker issued > 500 certificates for widely-known websites

..com	login.live.com	*.digicert.com
..org	login.yahoo.com	*.startssl.com
*.android.com	*.google.com	*.aol.com
*.globalsign.com	www.facebook.com	www.mossad.gov.il
*.mozilla.org	twitter.com	www.cia.gov
*.skype.com	*.windowsupdate.com	*.microsoft.com
*.torproject.org	www.update.microsoft.com	*.thawte.com
*.wordpress.com	addons.mozilla.org	…

- All certificates forged legitimate OCSP revocation information
  - Browsers asked real OCSP servers for revocation info of rogue certificates
  - Enabled DigiNotar to trace back certificate usage to mostly Iran

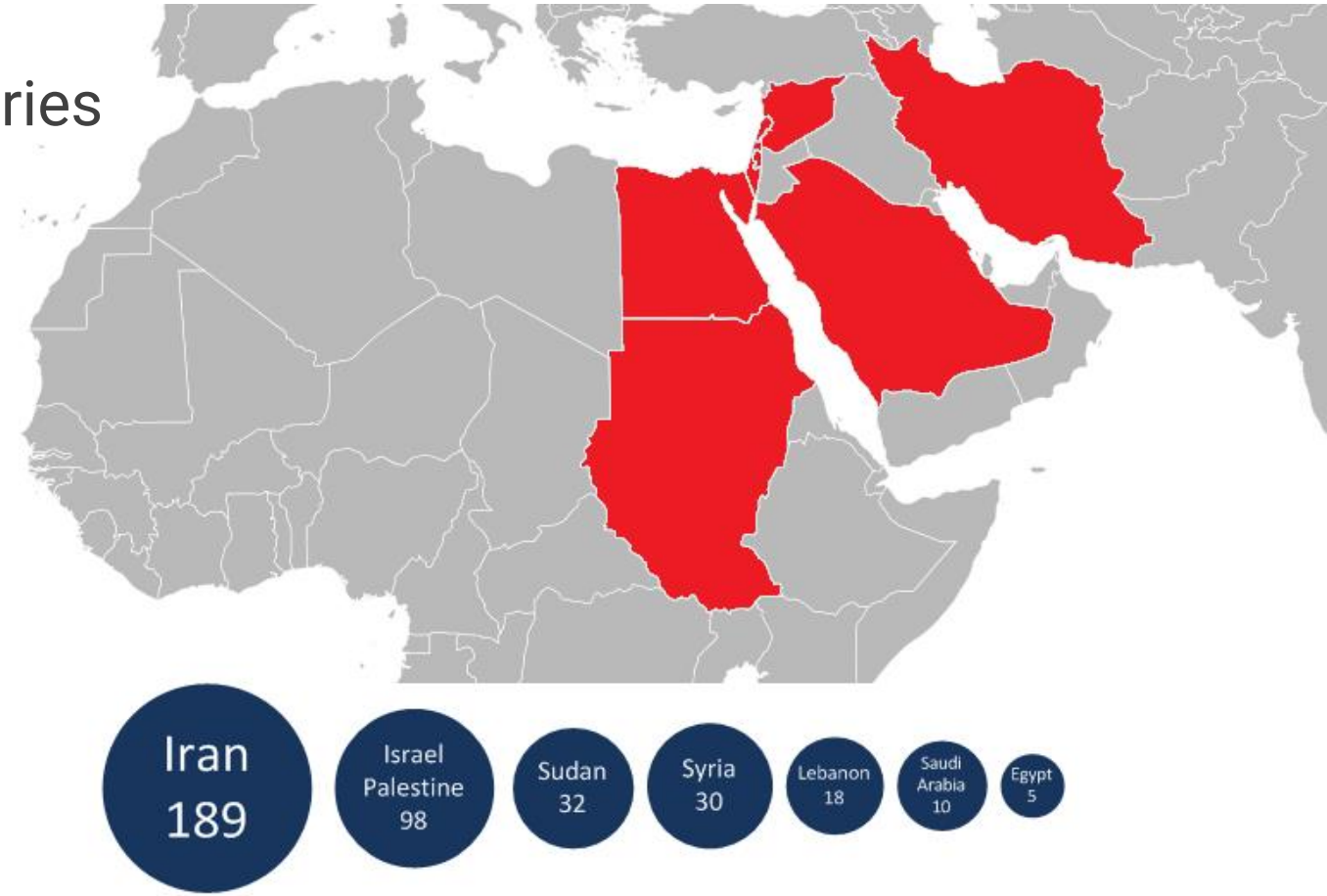
Flame Malware

2012: Highly-advanced malware found

Source: https://goo.gl/evDHnZ

- Found in several Middle East countries
  - Iran, Israel, Sudan, Syria, Egypt, …
  - Active for two years!

- Fairly complex
  - Over 20 attack modules
  - Network sniffing, microphone activation, file retrieval, …



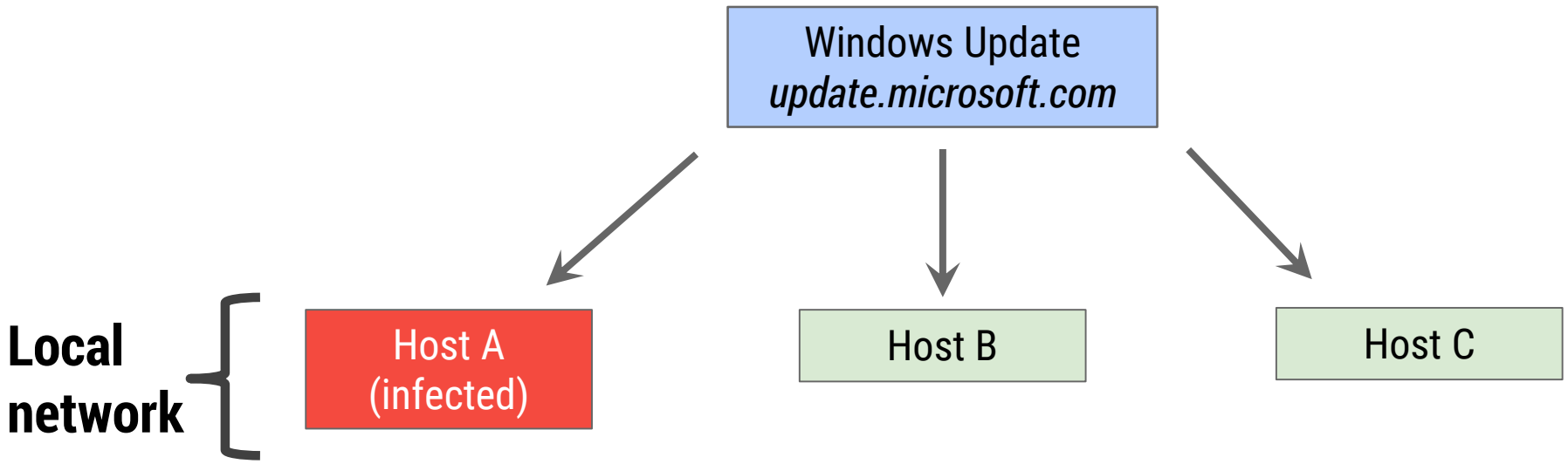
Most interesting aspect:
*Used a **valid** certificate, **not signed** by a CA!*

Flame – Windows Update

Assume PC has been infected via USB stick...

- How to attack other hosts in same network?
- Via direct remote exploits?
  - What if OS is not specifically vulnerable?

→ *Attack something that each Windows PC has: Update functionality*



Flame – Windows Update

Get hosts to download Update from attacker‘s server

Strategy

- Within LANs we often need proxies to allow browser to access Internet
- Manual configuration takes a lot of time

→ Web Proxy Auto Discovery Protocol (WPAD) invented

Idea

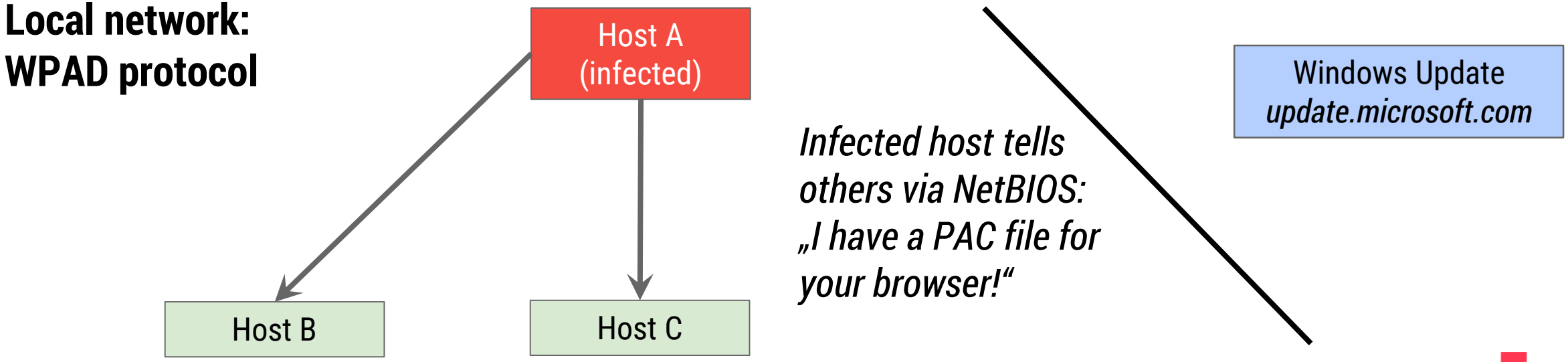
Distribute proxy information to browsers using PAC (Proxy Auto Config) file format

```
function FindProxyForURL(url, host) {
   // our local URLs from the domains below example.com don't need a proxy:
   if (shExpMatch(host, "*.example.com"))
   {
      return "DIRECT";
   }

   // URLs within this network are accessed through
   // port 8080 on fastproxy.example.com:
   if (isInNet(host, "10.0.0.0",  "255.255.248.0"))
   {
      return "PROXY fastproxy.example.com:8080";
   }

   // All other requests go through port 8080 of proxy.example.com.
   // should that fail to respond, go directly to the WWW:
   return "PROXY proxy.example.com:8080; DIRECT";
}
```

Flame – Windows Update

How WPAD works...

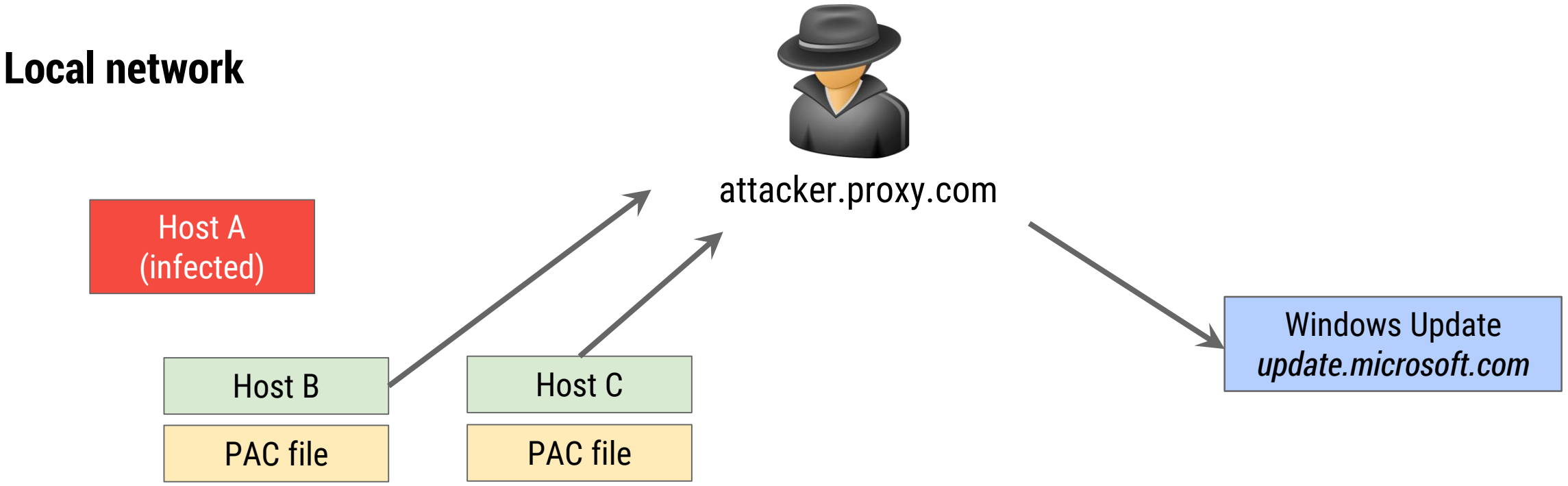
- Host asks via DHCP, DNS and NetBIOS protocols for file wpad.dat
- E.g., find proxy via NetBIOS (= address resolution in MS protocol world)
  - Name resolution using broadcasts, infected host answers



Flame – Windows Update

Hosts have received PAC file with content
"For update.microsoft.com use attacker.proxy.com"

→ Now hosts ask Windows Update for news



Flame – Windows Update

Now, not-yet infected hosts go to update.attacker.com for Windows Updates.

We want to distribute Flame via Windows update file

Q: Doesn‘t update.microsoft.com use HTTPS?

A: It also accepts HTTP :-)

Q: Doesn‘t Microsoft sign updates using special Code Signing certificates?

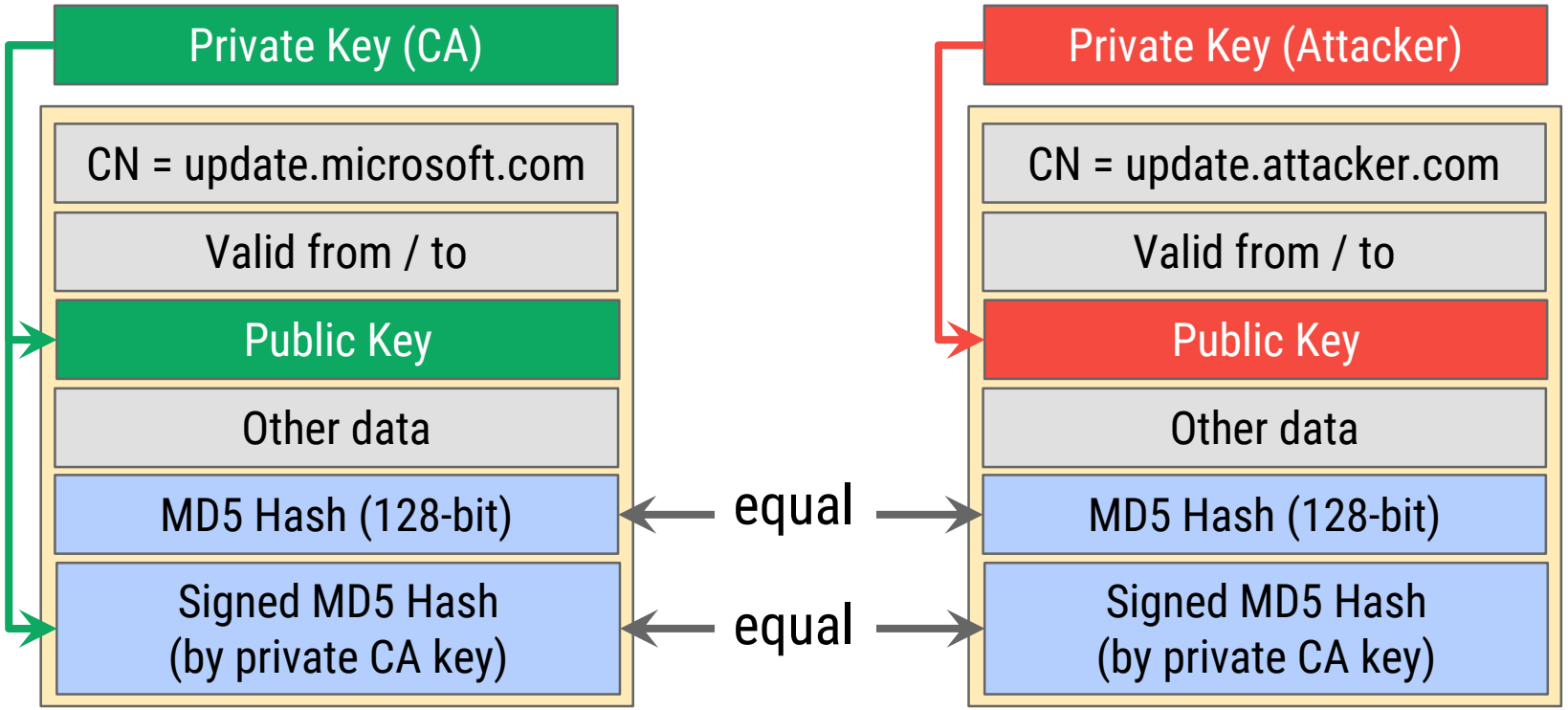
A: There was an old CA that used MD5 signatures for certificates

→ MD5 is broken since many years - cryptographic attack possible!

→ Quite easy to find collision, create certificate and make it seem to be signed by the CA

Flame – Windows Update

- *Attacker used attack on MD5 to create valid certificate*
  - *Most interesting thing: New, so far unknown attack method used*
  - *„Chosen-prefix collision attack“ → very timing and cost-intensive collision search*



- *Attacker signed malware (Windows Update package) with own private key*
- *Windows happily installed the malware*

Improvements

- Public Key Pinning (RFC 7469)
  - Addresses problem that any CA can issue any certificate
  - Enables site owners to explicitly specify legitimate fingerprints

- DANE (RFC 6698)
  - Based on DNSSEC (integrity checking for DNS zones)
  - Alternative approach for pinning

- Certificate Transparency (CT) See: https://goo.gl/2kqVTT
  - Framework to audit and monitor certificates → quickly find fraudulent certificates
  - Legitimate CAs shall submit certificates to CT logs
  - If unknown certificate used for site, warn client

Implementation Attacks

Overview

Problem

Those who design protocols / ciphers are often not those who implement it

- Many have critical conceptual flaws ***but*** even more „bad code“ is out there
- Bypassing (strong) crypto is always easier than breaking it…

Types

- On-purpose interception of encrypted traffic
  - Install root certificate in user‘s browser and perform MITM attack

- Certificate Validation Flaws
  - Library issues
  - Wrong usage of APIs

Dell does a Superfish, ships PCs with easily cloneable root certificates

Root certificate debacle that hit Lenovo now visits the House of Dell.

by **Dan Goodin** - Nov 23, 2015 6:40pm CET

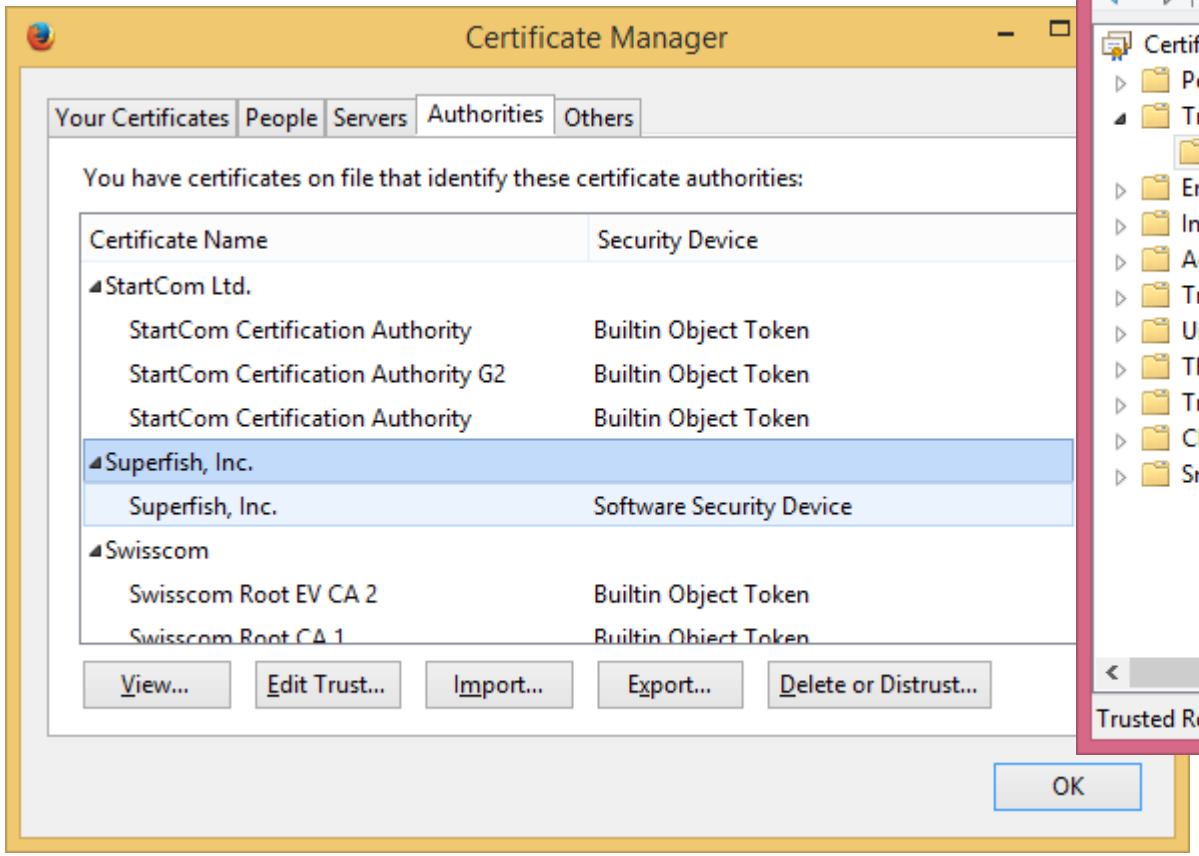
222

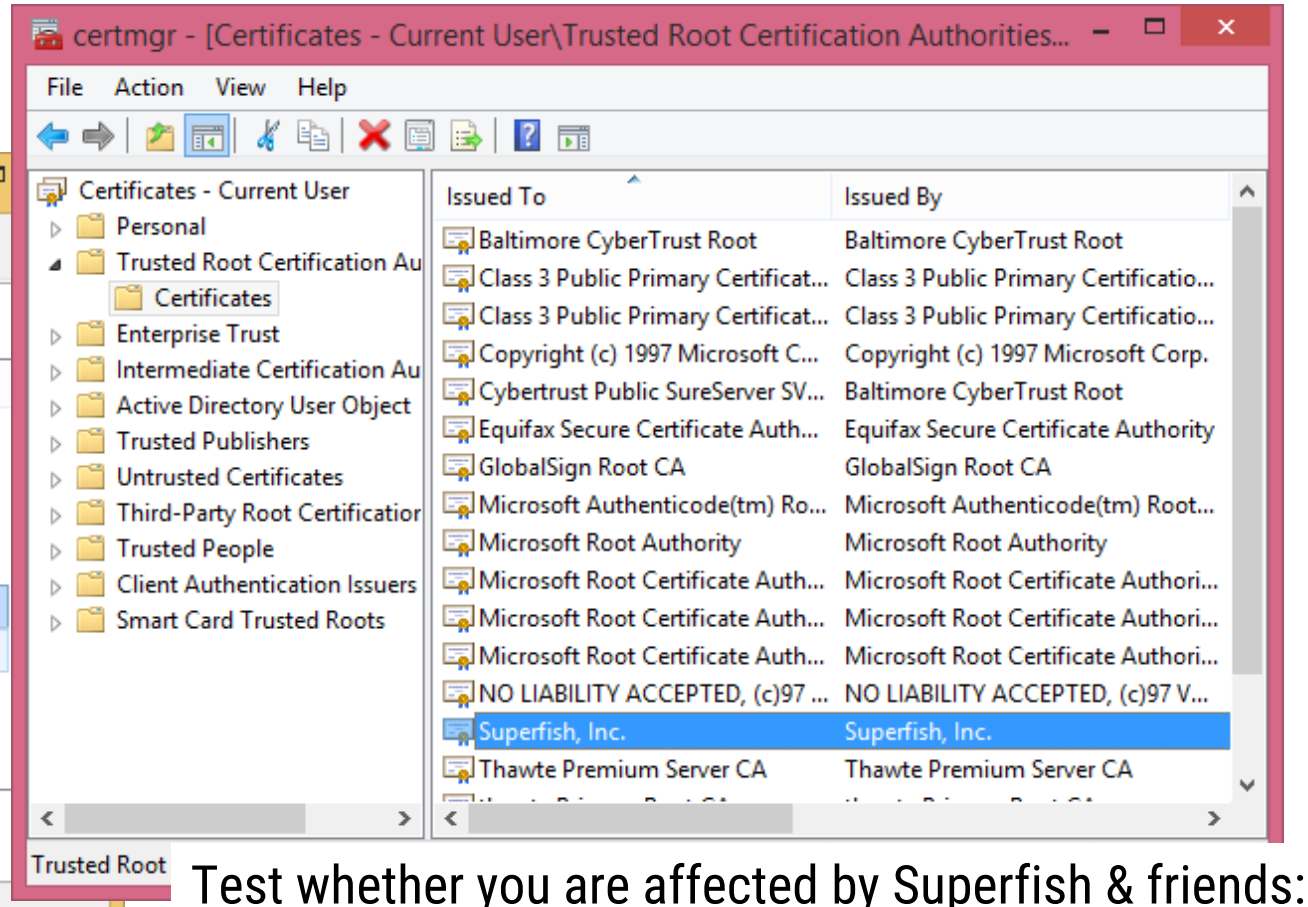
In a move eerily similar to the Superfish debacle that visited Lenovo in February, Dell is shipping computers that come preinstalled with a digital certificate that makes it easy for attackers to cryptographically impersonate Google, Bank of America, and any other HTTPS-protected website.

Source: http://goo.gl/esDjtS

- Products want to manipulate even encrypted web traffic
  - „Enterprise“ security products, Antiviruses, Ad-Blockers, Adware, …
- „Superfish“
  - Analyzes images on webpages and provides matching ads
  - Preinstalled on many Lenovos
  - Became public in 02/2015
  - Same issue with others, e.g. on Dell notebook (11/2015)

Superfish





Test whether you are affected by Superfish & friends:
https://superfish.tlsfun.de/
https://badssl.com/dashboard/

→ Wouldn‘t Certificate Pinning (HPKP) uncover such a MITM attack?

*Sad truth: No, because **manually** installed root CAs disable pinning in browsers!*

See: https://goo.gl/OZksTj and https://goo.gl/teWRTO

IAIK TU Graz

Certificate Validation Flaws

How to check certificates correctly?

1. Ensure server certificate corresponds to intended domain name

2. All chain certificates must be checked that
   - They have not expired
   - Their signatures are valid

3. Foreach intermediate certificate check
   - What key usage is allowed, e.g. sign certificates for web but not Code Signing
   - That it can be used to sign the hostname in the leaf certificate

How to do it correctly with OpenSSL? See https://goo.gl/qbFDZw

Aging and bloated OpenSSL is purged of 2 high-severity bugs

Padding oracles and memory corruption threats caused by use of older schemes.

by **Dan Goodin** - May 3, 2016 5:50 pm UTC

50

The ASN.1 implementation in OpenSSL before 1.0.1o and 1.0.2 before 1.0.2c allows remote attackers to execute arbitrary code or cause a denial of service (buffer underflow and memory corruption) via an ANY field in crafted serialized data, aka the "negative zero" issue.

Source: http://goo.gl/JElv4d

Recurring problems in widely-used libraries (OpenSSL, GnuTLS, Microsoft Crypto API, …)

- **Padding oracle**

Allow attacker to repeatedly probe encrypted payload for clues about plaintext inside

- **Memory corruption**

Code execution using malformed digital signatures

Heartbleed

Information disclosure vulnerability in OpenSSL

→ Exploits faulty implementation of „Heartbeat“ protocol

How?

- Developer forgot to check length of input variable
- Attacker may request up to 64 KB of server process memory

Consequences: Leak of private session information

- Response could include session cookies, passwords, etc.
- Private keys
  - Simple but slow: Search for prime numbers in Heartbleed message
  - If one prime found → enough to calculate private RSA key

See: https://goo.gl/PC9liK

goto fail;

About the security content of iOS 7.0.6

Impact: An attacker with a privileged network position may capture or modify data in sessions protected by SSL/TLS
Description: Secure Transport failed to validate the authenticity of the connection. This issue was addressed by restoring missing validation steps.

Source: http://goo.gl/xQQTkU

What went wrong?

```
static OSStatus SSLVerifySignedServerKeyExchange(SSLContext *ctx,
bool isRsa, SSLBuffer signedParams, …) {
...
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
  goto fail;
  goto fail;
…
err = sslRawVerify(ctx, ctx->peerPubKey, dataToSign, dataToSignLen,
                   signature, signatureLen);
...
}
```

Source: http://goo.gl/iK8FbN

- Code will always jump to fail after second goto fail;
- Skips call to sslRawVerify
  - Intended signature verification will never be executed

→ *Any* private key will be accepted!

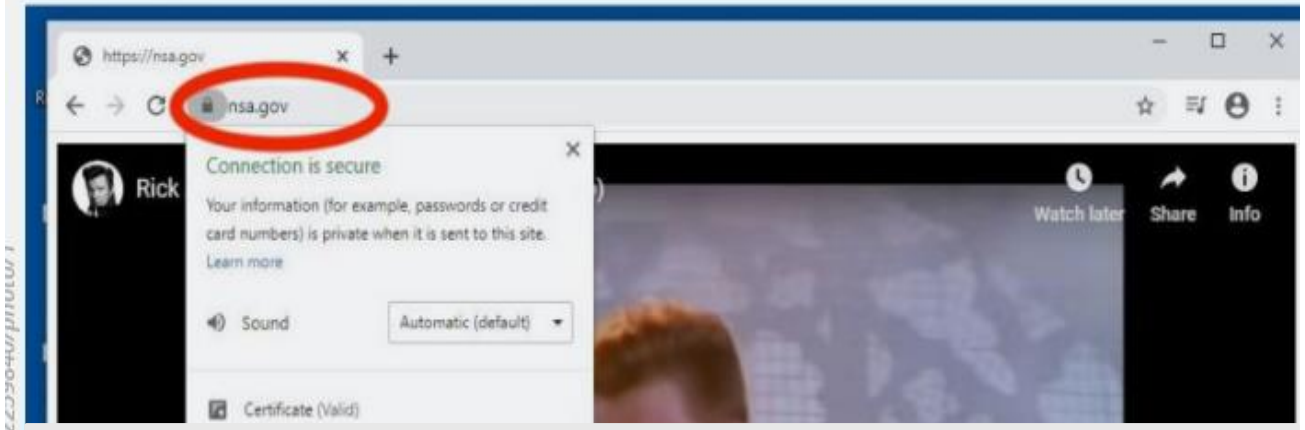
See: https://goo.gl/VYJDDk

GOT CERT VALIDATION? —

Critical Windows 10 vulnerability used to Rickroll the NSA and Github

Attack demoed less than 24 hours after disclosure of bug-breaking certificate validation.

DAN GOODIN - 1/16/2020, 1:30 AM



A spoofing vulnerability exists in the way Windows CryptoAPI (Crypt32.dll) validates Elliptic Curve Cryptography (ECC) certificates. Attackers can exploit it by using a spoofed code-signing certificate to sign a malicious executable. A successful exploit could also allow the attacker to conduct man-in-the-middle attacks and decrypt confidential information on user connections to the affected software.

NSA found a certificate validation bug (CVE-2020-0601) concerning ECDSA signatures

- **What's the problem?**

CryptoAPI's CA certificate cache falsely thinks a fake root CA is also part of the CA certificate store as soon as its public key and serial number match a certificate that is already in in the cache

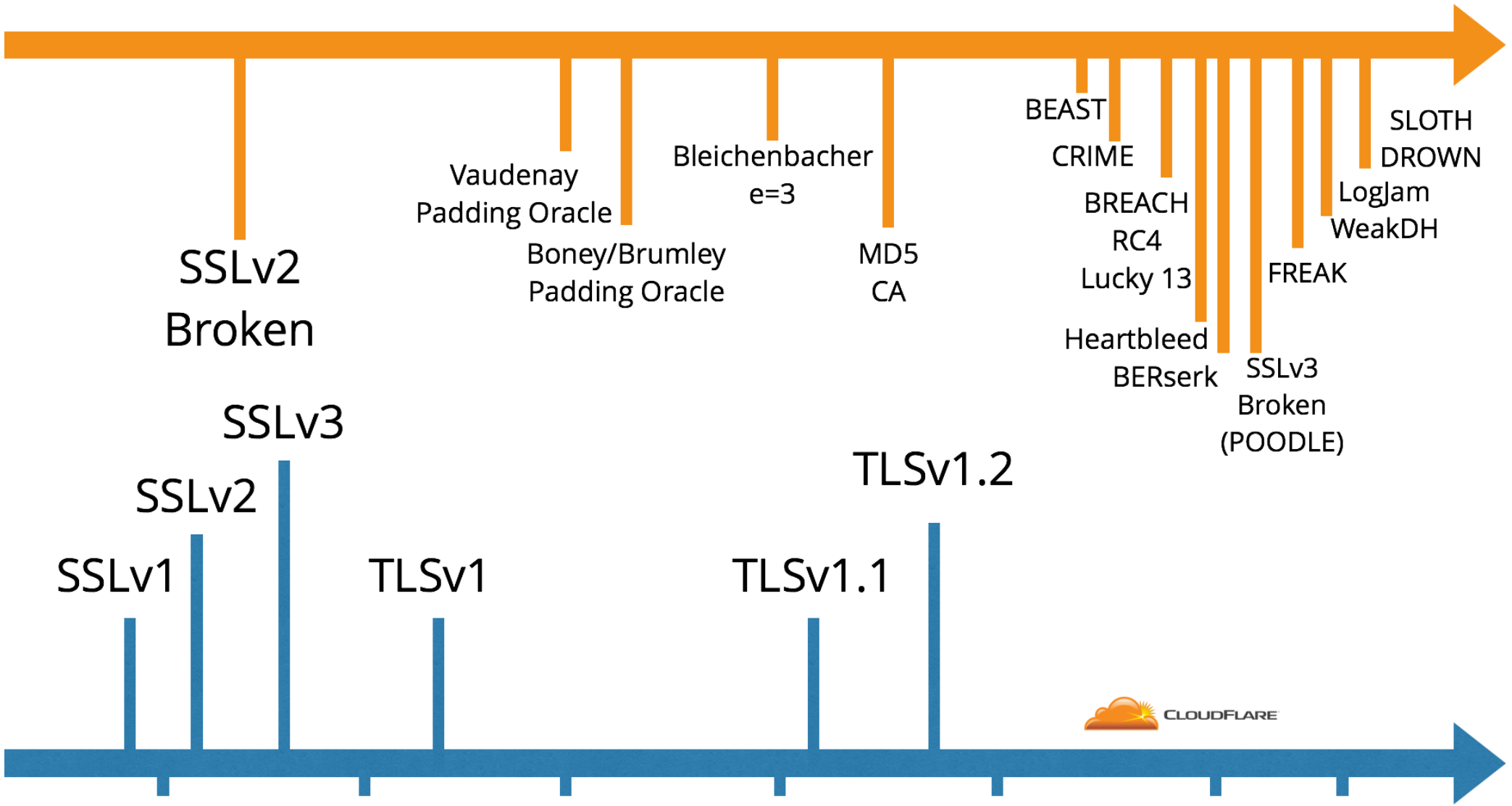
- **Consequence?**

Attackers can spoof trusted ECC root certs by crafting valid private keys by just copying public key + all used cert parameters

Windows forgets to check the base point generator G'.

Protocol Attacks

Overview



Source: Cloudflare

Overview

Past attacks in categories…

- Downgrade attacks: *Freak, Logjam*
- Compression attacks: *Crime, Time, Breach*
- Attacks via Padding Oracles: *Lucky 13, Beast, Poodle*

See: https://goo.gl/vKwCm4

- RSA-related attacks: *Bleichenbacher, Drown*
- Insecure Renegotiation

Downgrade Attacks

Problem

- For compatibility reasons, weak ciphers often remain activated
- Attacker could trick server & client into negotiating connections using them

Example: „Factoring RSA Export Keys“ (FREAK) - 2014

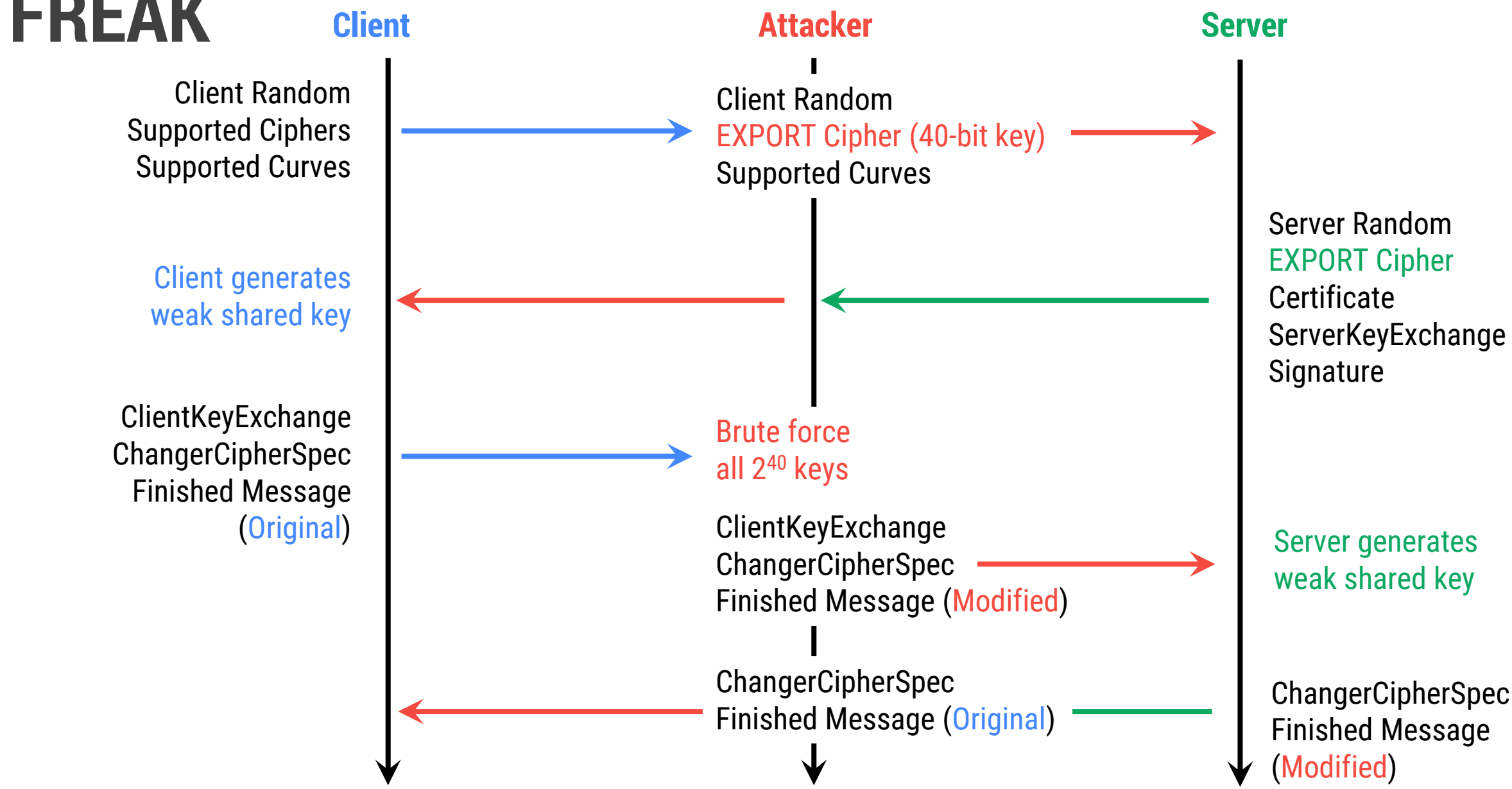
1. Perform MITM attack
2. Swap supported ciphers with EXPORT cipher (= weak encryption key)
3. Crack EXPORT key

EXPORT suites

```
RSA-EXPORT-WITH-RC4-40-MD5
RSA-EXPORT-WITH-DES40-CBC-SHA
DHE-DSS-EXPORT-WITH-RC4-56-SHA
```

→ *512-bit RSA key, 40-bit RC4 key*

FREAK



All traffic beyond this point is encrypted with weak shared keys. Attacker can read/modify everything!

Compression Attacks

= „Message length side channel“ / „Compression Oracle“

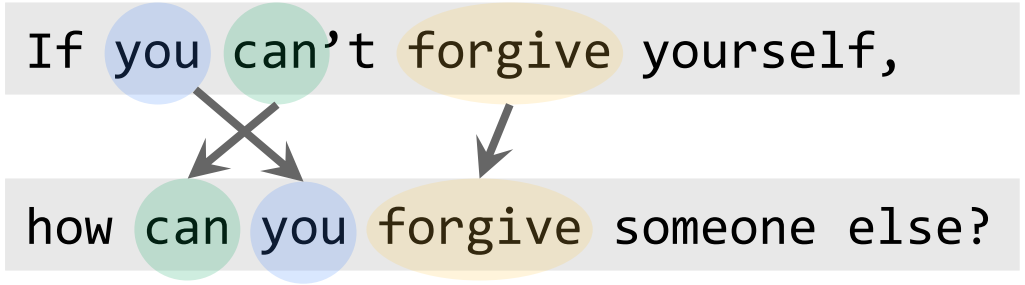
Problem

- If server applies compression on encrypted data,
  attackers may add own data which is then also compressed
- Size of compressed content lets you draw conclusions on content

How does it work?

- Compression algorithms eliminate redundancy → repeated characters
- If size of compressed content is **reduced despite** appending bytes to encrypted msg
  → Attacker can assume:
  Injected content matches some part of unknown source part, he tries to find out!

Compression Attacks



Compression would keep only one copy of duplicated data

„Oracle“ exists if attacker can add arbitrary data, compressed in same way as some unknown secret data. Now by observing size of compressed output
→ if output size reduced by compression, guess was correct

Example

```
GET /JSESSIONID=X HTTP/1.1
Host: www.example.com
Cookie: JESSIONID=B3DF4B07AE33CA
```

Injected data X → incorrect guess: 73 bytes compressed

```
GET /JSESSIONID=B HTTP/1.1
Host: www.example.com
Cookie: JESSIONID=B3DF4B07AE33CA
```

B → correct guess: 72 bytes compressed

DROWN Attack

= Decrypting RSA with Obsolete and Weakened Encryption

Ingredients

- Practical attack against SSLv2
- Same certificate shared among used protocols (SSLv2, …, TLS 1.2)
- Implementation errors in OpenSSL
  - Non-standard compliant SSLv2 client can force handshake
  - EXPORT ciphers do keep some bits unencrypted

Consequence?

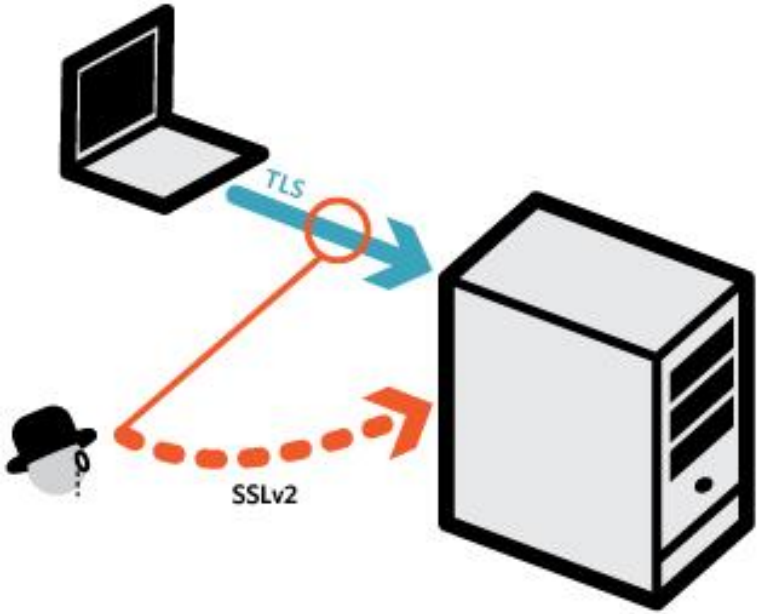
Enables MITM attacks where attacker can decrypt session keys

See https://drownattack.com

DROWN Attack

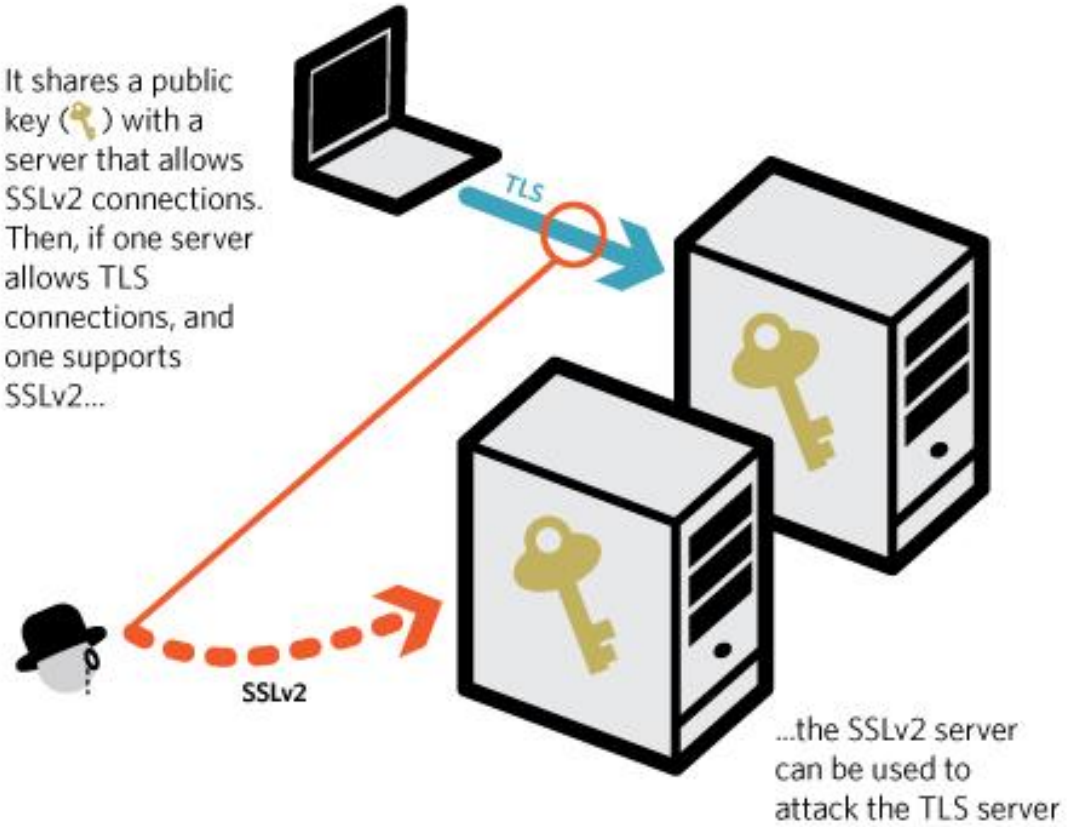
A server is vulnerable to DROWN if:

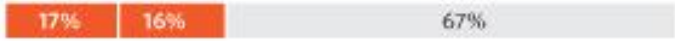
It allows both TLS and SSLv2 connections



17% of HTTPS servers still allow SSLv2 connections

It shares a public key (🔑) with a server that allows SSLv2 connections. Then, if one server allows TLS connections, and one supports SSLv2...





When taking key reuse into account, an additional 16% of HTTPS servers are vulnerable, putting 33% of HTTPS servers at risk

Source: https://goo.gl/nuhsfT

Lessons Learned?

- X.509 certificate handling & ASN.1 parsing are hard to implement ***correctly***
- Before breaking SSL / TLS protocols
  →„*cheaper*“ to attack PKIs or exploit implementation flaws

- On-purpose TLS interception is dangerous, nobody gets it right

See: https://goo.gl/teWRTO

- If attackers can identify only **one** bit of information, it is over
  - „Brute-Forcing“ via Compression, Padding, or Timing Oracles
- Enabled support for weak / insecure ciphers and protocols for compatibility poses serious risks → *Downgrade attacks*

DNS Security

DNS Issues

Once upon a time…

DNS was designed for a closed environment (ARPAnet) – this changed obviously…

Attacking DNS servers

- Denial-of-Service attacks
  - Make it unavailable! What happens if a DNS server cannot be reached? :-)
- DNS Amplification attack
  - Multiply amount of traffic flood thanks to „large replies“ after „small queries“
- Cache Poisoning
  - Let user connect to wrong destination IP address

Spamhaus DDoS grows to Internet-threatening size

More than 300 Gb/s of traffic aimed at the anti-spam site's hosting.

by **Peter Bright** - Mar 27, 2013 8:30pm CET

258

Source: http://goo.gl/czi8eZ

What happened?

DNS Amplification Attack producing 300 Gbit/s traffic on spamhaus.org

Why is this problematic?

- 300 Gb/s is the scale that threatens the Internet‘s core routers (Tier-1)
- By overloading them, you risk breaking global connectivity

Remedy?

Well, Anycasting helps, more or less…

DNS Amplification Attack

Idea: Amplify the bandwidth you can use for a DDoS attack

Ingredients

1. Being able to set an arbitrary (spoofed) source IP address,
   e.g. via ICMP / UDP as they require no handshake
2. Make response to query significantly *larger* than the request
3. Apply this operation distributed using „Open DNS resolver“
   → Servers that resolve recursive DNS requests for anyone on Internet

Why does it work?

E.g. attacker sends query with e.g. 60 bytes, response has 3000 bytes
→ Traffic amplification factor of 50

→ Attacker queries with 100 Mbit/s, responses produce 5 Gbit/s !!!

DNS Security

Scenario

You go to a café and use their WiFi
→ How does your browser find www.google.at?

Mostly like this…

- Ask local name server, obtained via DHCP
  - You implicitly trust this server!

- Can return **any** answer for google.at,
  including a malicious IP address that acts as Man-in-the-middle
  - Think of captive portals / hotspot login pages that arise after connecting…

→ How can you know you are getting the „correct“ response? :-)

Cache Poisoning

Scenario

- Assume you control the DNS zone evil.at
- You receive a query for www.evil.at and reply

```
;; QUESTION SECTION:
;www.evil.at.                   IN      A

;; ANSWER SECTION:
www.evil.at.            3600    IN      A       72.52.4.90

;; AUTHORITY SECTION:
evil.at.                600     IN      NS      ns1.evil.at.
evil.at.                600     IN      NS      google.at.

;; ADDITIONAL SECTION:
google.at.              5       IN      A       72.52.4.90
```

Glue record pointing to attacker's IP, not Google's!

And it gets cached!

Cache Poisoning

Problem

How do you get a victim to look up evil.at?

For the attack to work, your forged DNS entry has to be fetched…

One possible solution

- You might connect to their mail server and send `HELO www.evil.at`
- The mail server will look up to check if it corresponds to the connecting IP address (SPAM filtering)

→ *While resolving, you also learn the tainted DNS record*

Mitigation?

Only accept glue records from the domain you asked for…

Defense

Q: How to protect against forged oder manipulated DNS data?

A: DNS Security Extensions (DNSSEC)

How?

- Chain-of-Trust between all involved name servers
- DNSSEC-enabled servers digitally sign all their answers cryptographically
  - Use (secret) private RSA key for signing
- DNS resolvers verify if signature matches received records
  - Public RSA key for verification is normal RR with type „DNSKEY“

Effect?

DNS content cannot be modified without being detected!

Outlook

Lecture exam!