

ADVENTURES IN TLS

VITALY SHMATIKOV

SSL/TLS

Secure Sockets Layer and Transport Layer Security protocols

- Same protocol design, different cryptographic algorithms

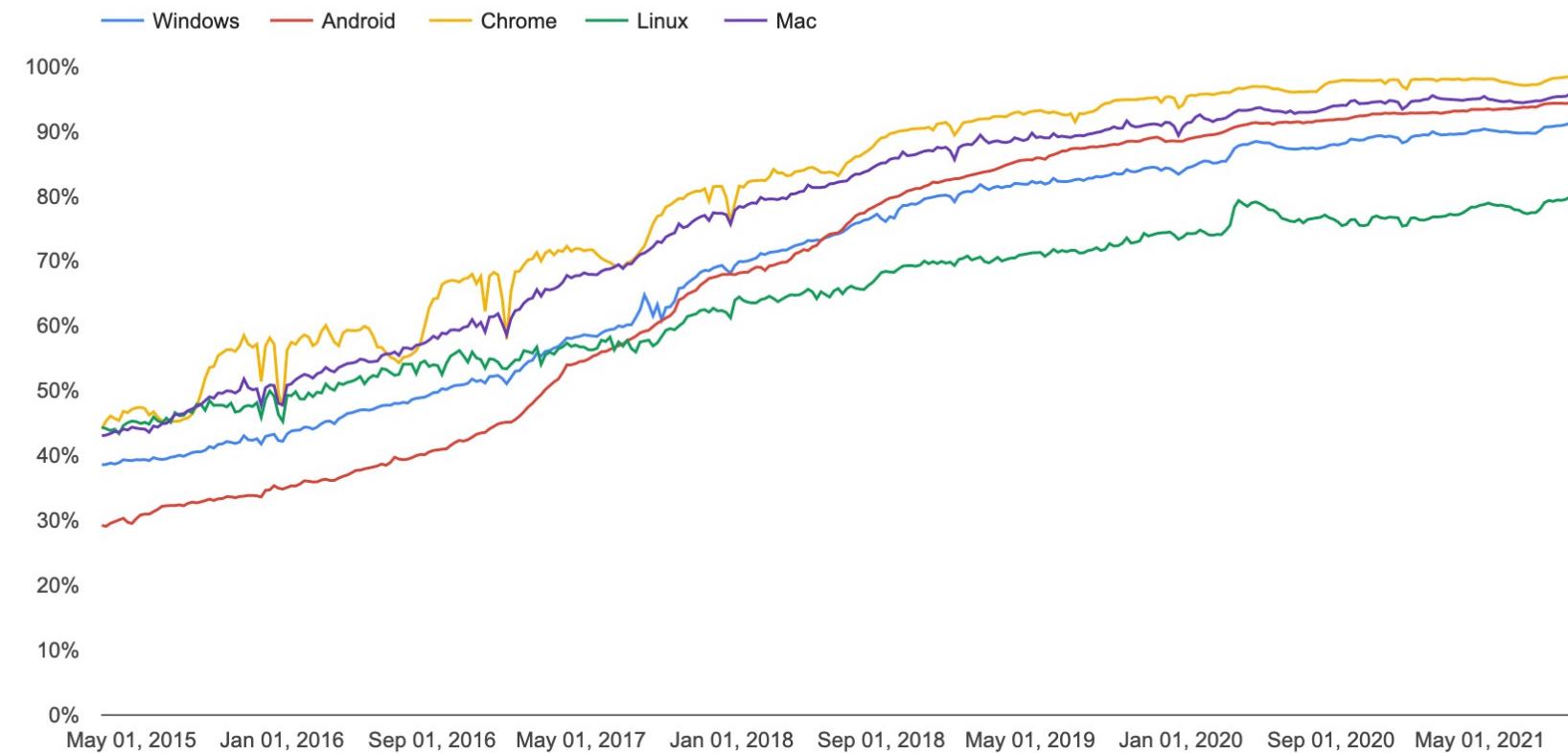
The de facto standard for Internet security

- “The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications”

Deployed in every Web browser (HTTPS); also mobile applications, payment systems, VoIP, many distributed systems, etc.

HTTPS Adoption by Websites

Percentage of pages loaded over HTTPS in Chrome by platform



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

SSL / TLS Guarantees

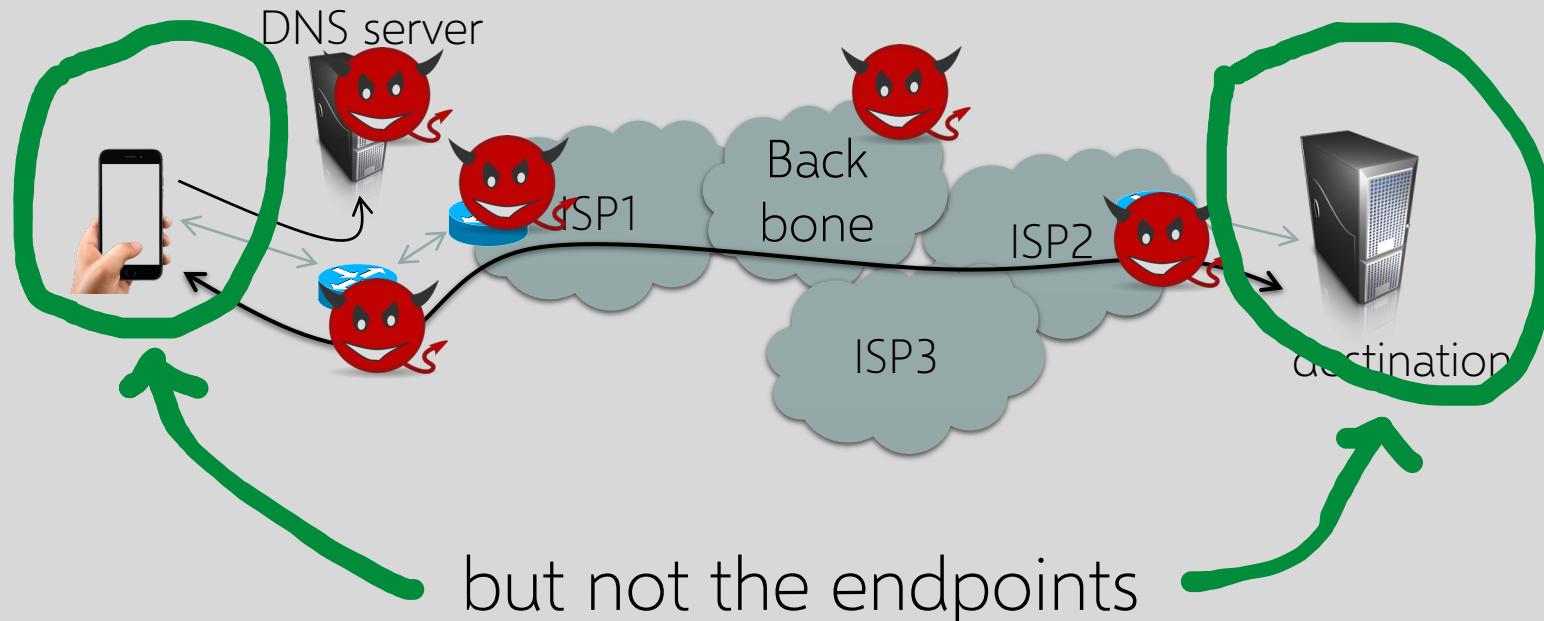
End-to-end secure communications in the presence of a network attacker

- Attacker completely Owns the network: controls Wi-Fi, DNS, routers, his own websites, can listen to any packet, modify packets in transit, inject his own packets into the network

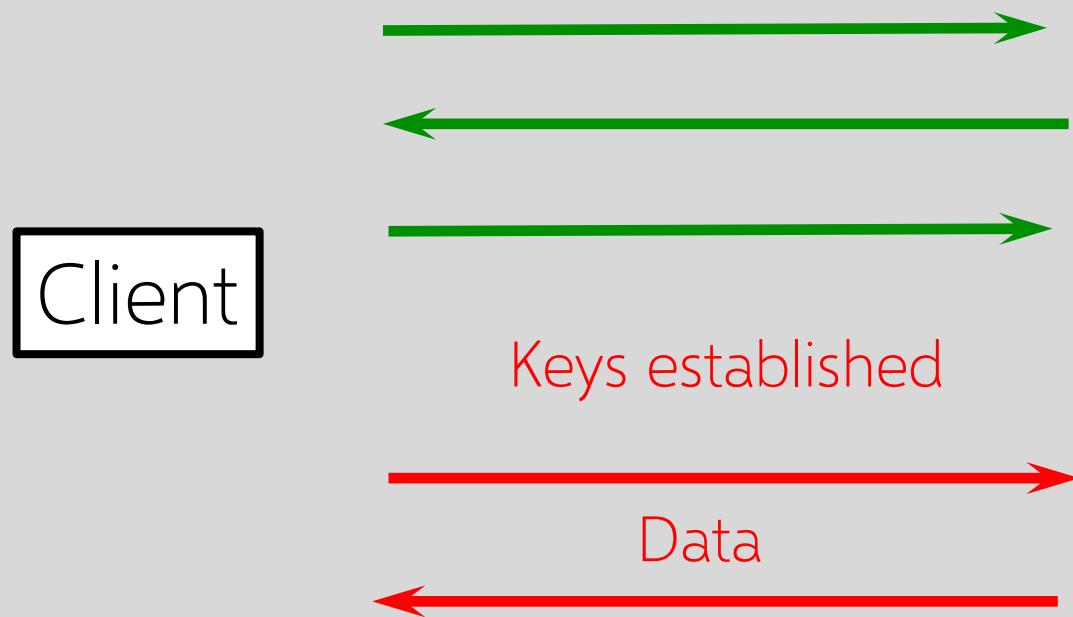
Scenario: you are reading your email from an Internet café connected via a r00ted Wi-Fi access point to a dodgy ISP in a hostile authoritarian country

TLS Threat Model

Remember TCP/IP, BGP, DNS attacks?
TLS is all that stands between us and oblivion...



Establishing a Secure Channel



Handshake protocol:
use public-key cryptography to authenticate each other, establish shared symmetric keys

Server

Record protocol:
use symmetric keys to protect confidentiality, integrity, authenticity of exchanged data

SSL/TLS Handshake Protocol

Negotiate version of the protocol and the set of cryptographic algorithms to be used

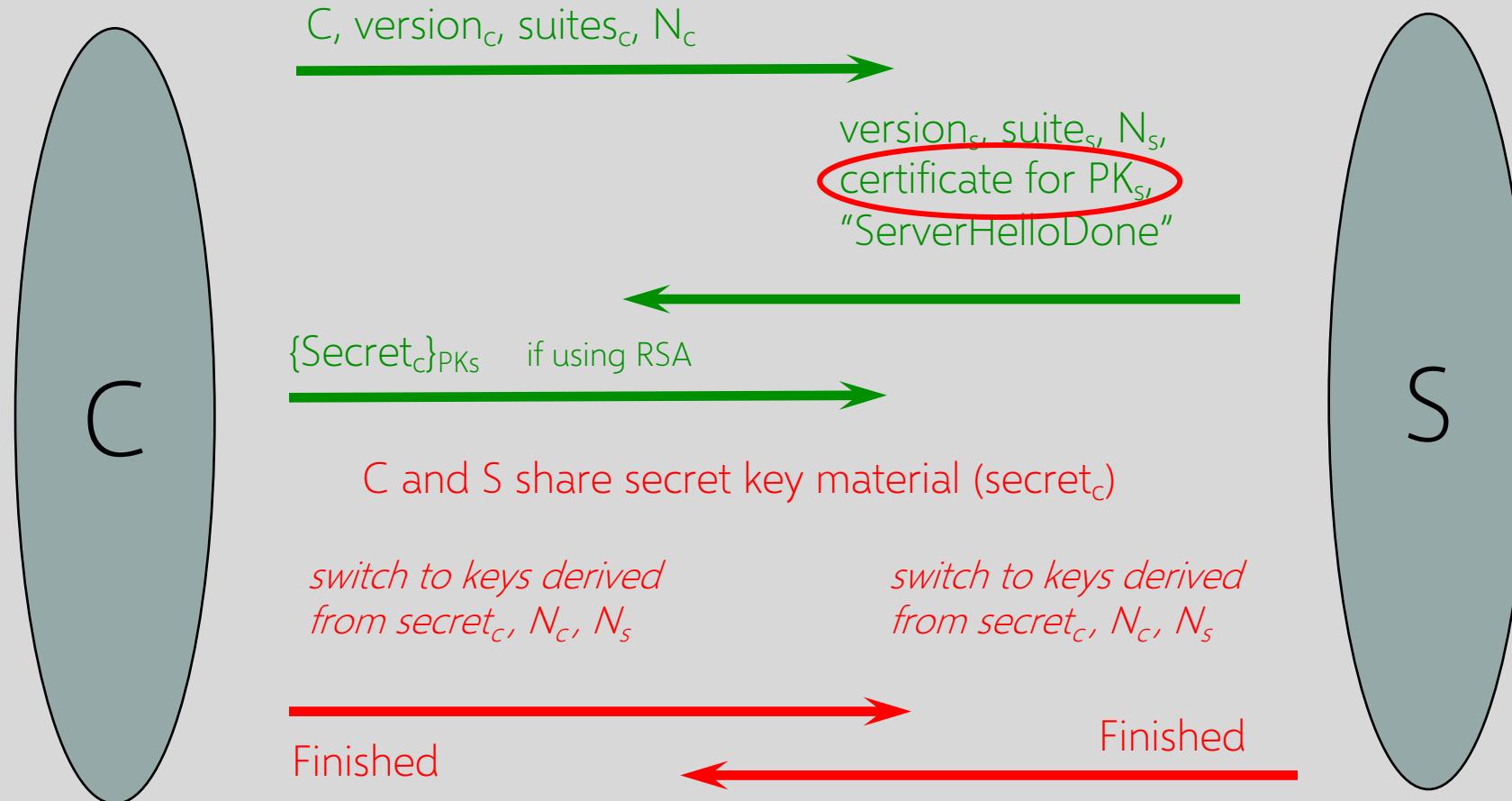
- Interoperability between different implementations

Authenticate server and client (optional)

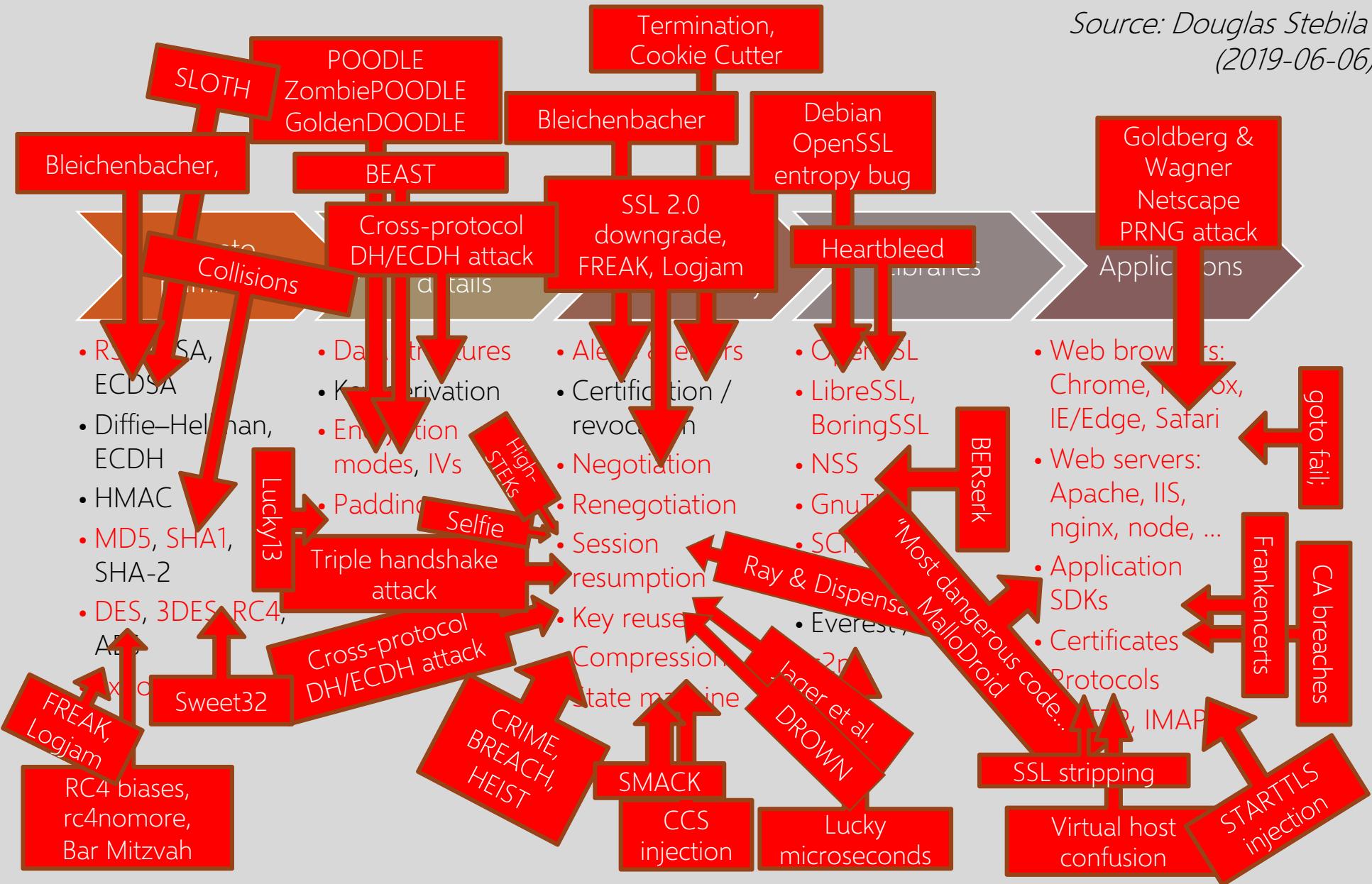
- Use digital certificates to learn each other's public keys and verify each other's identity
- Often only the server is authenticated

Use public keys to establish a shared secret

“Core” SSL/TLS Handshake

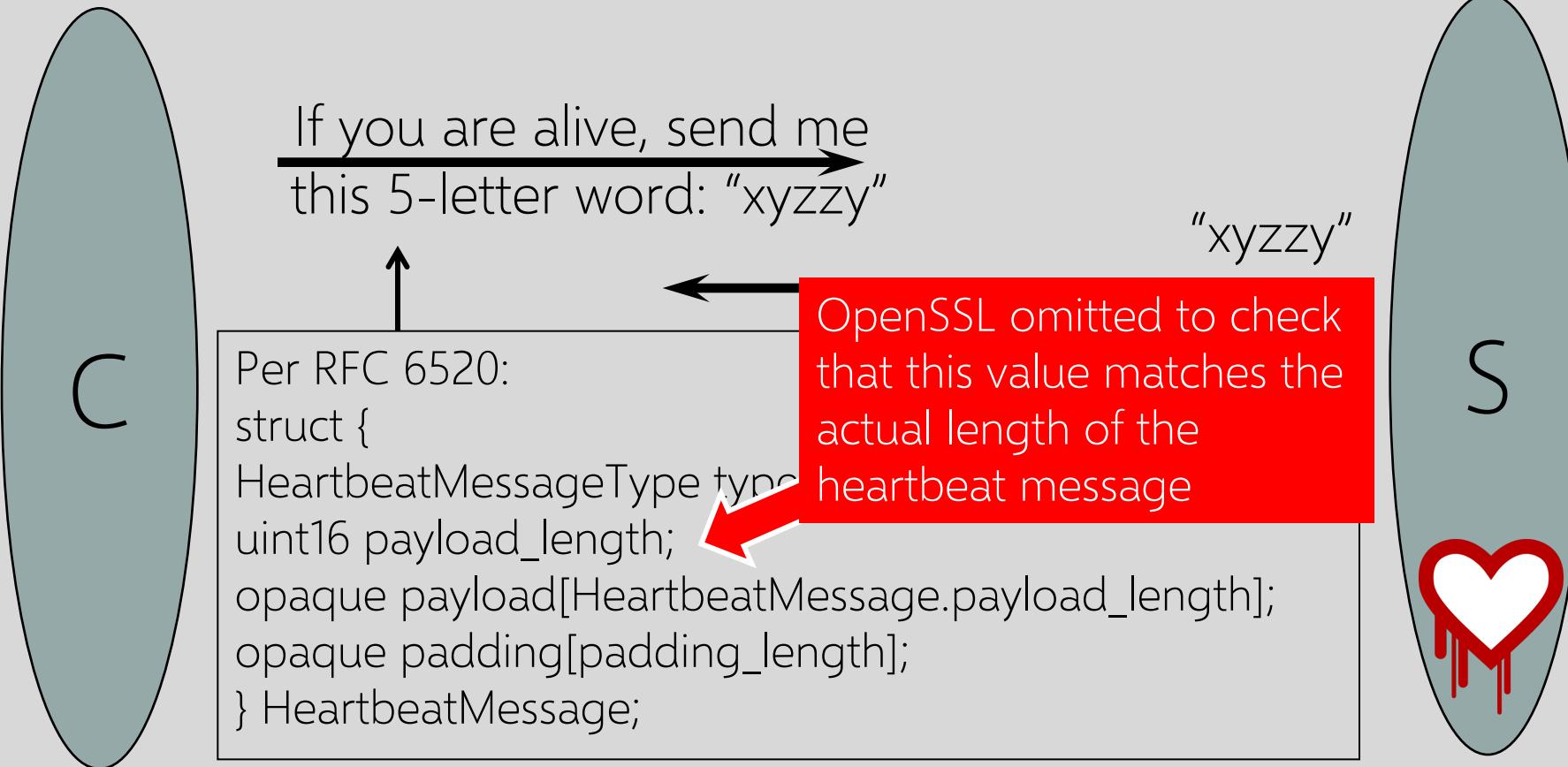


Attacks on TLS



TLS Heartbeat

A way to keep TLS connection alive
without constantly transferring data

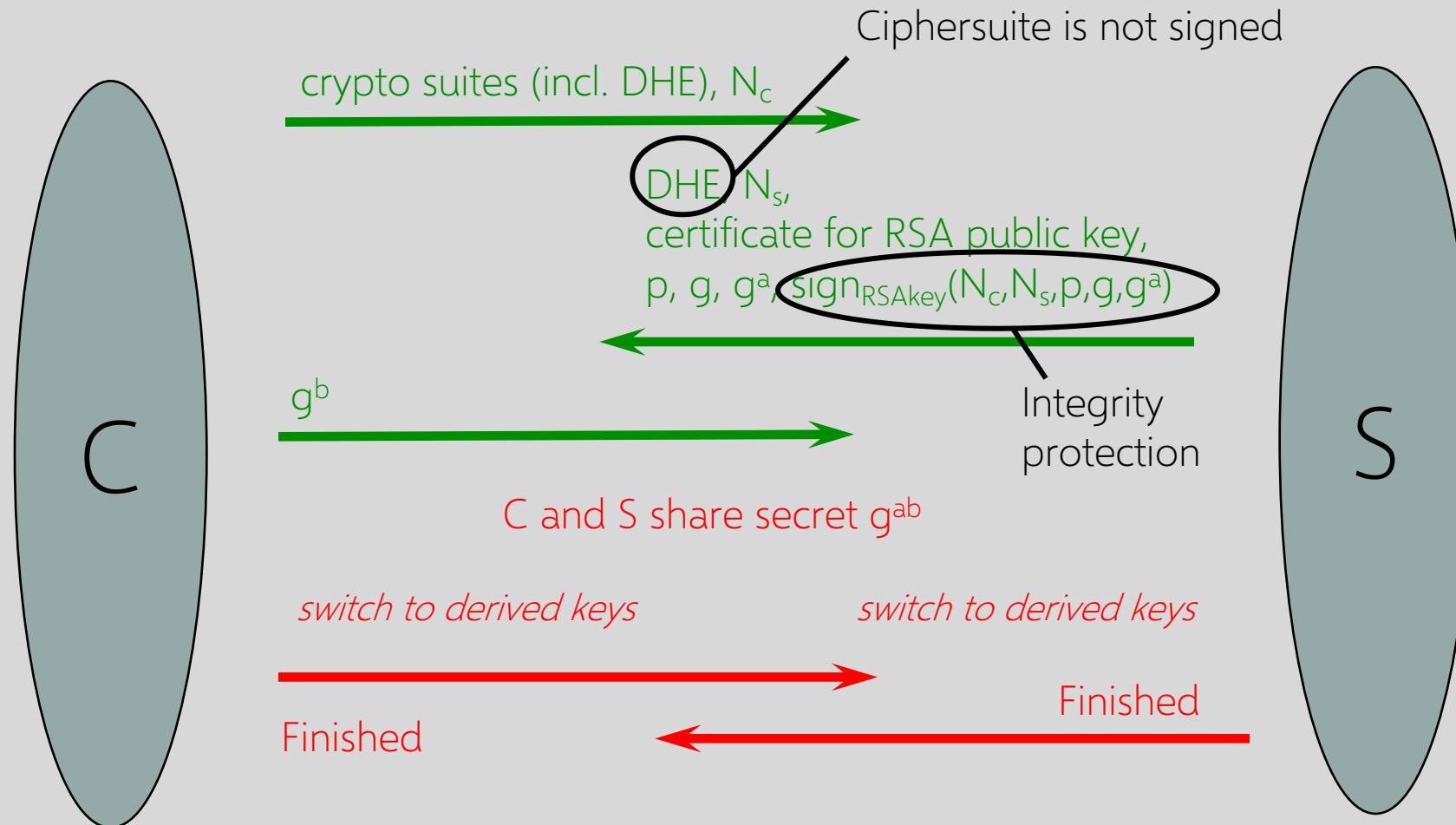




Heartbleed Consequences

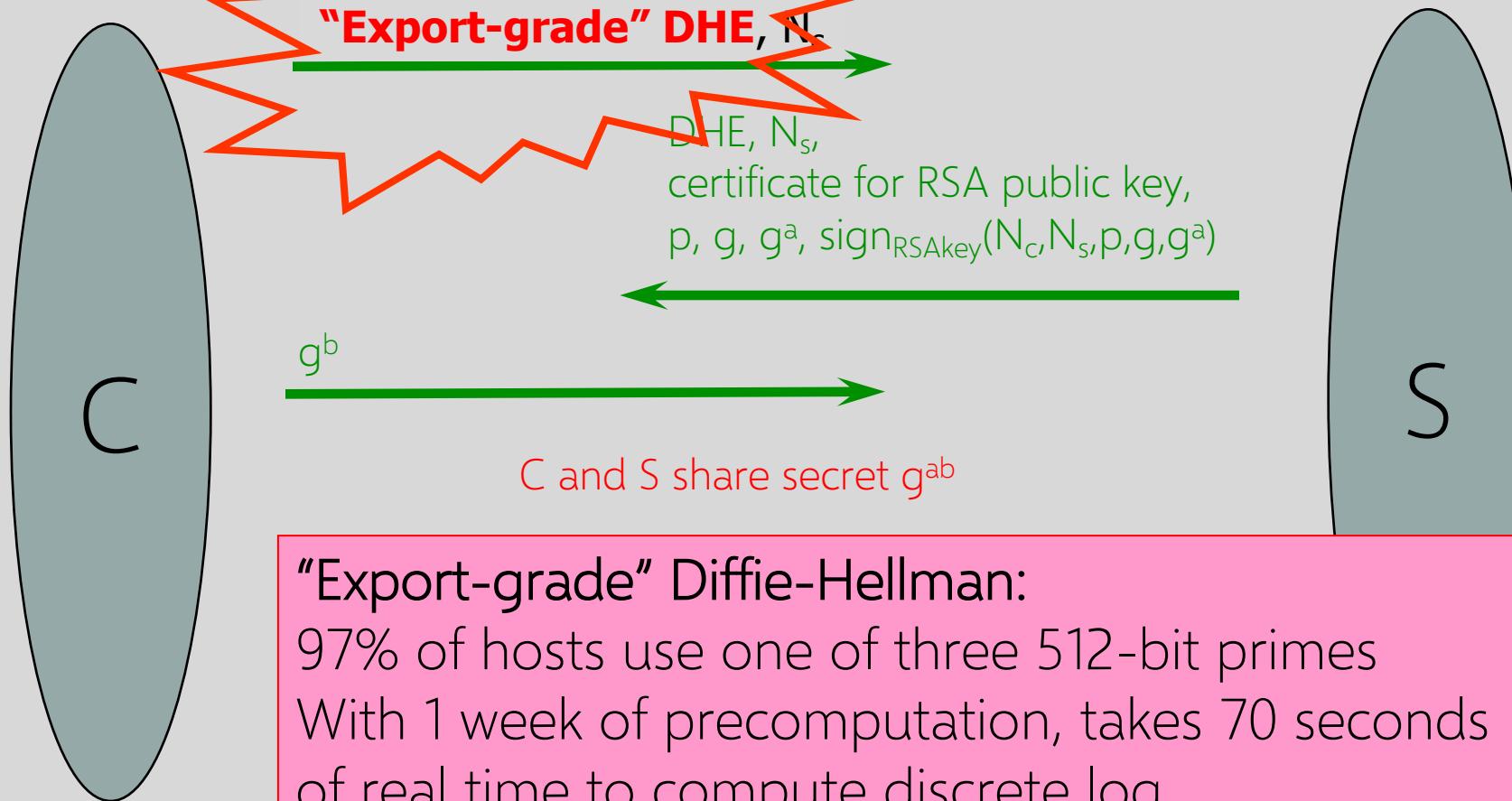
- Attacker can obtain chunks of server memory
 - >Passwords, contents of other users' communications, even the server's private RSA key
- Assisted by a custom allocator that does not zero out malloc'd memory (for "performance," natch!)

TLS with Diffie-Hellman



LOGJAM attack

DH Downgrade by MITM





LOGJAM SSL ATTACK

Warning! Your web browser is vulnerable to Logjam and can be tricked into using weak encryption.

BEAST attack

HTTP
Encrypted

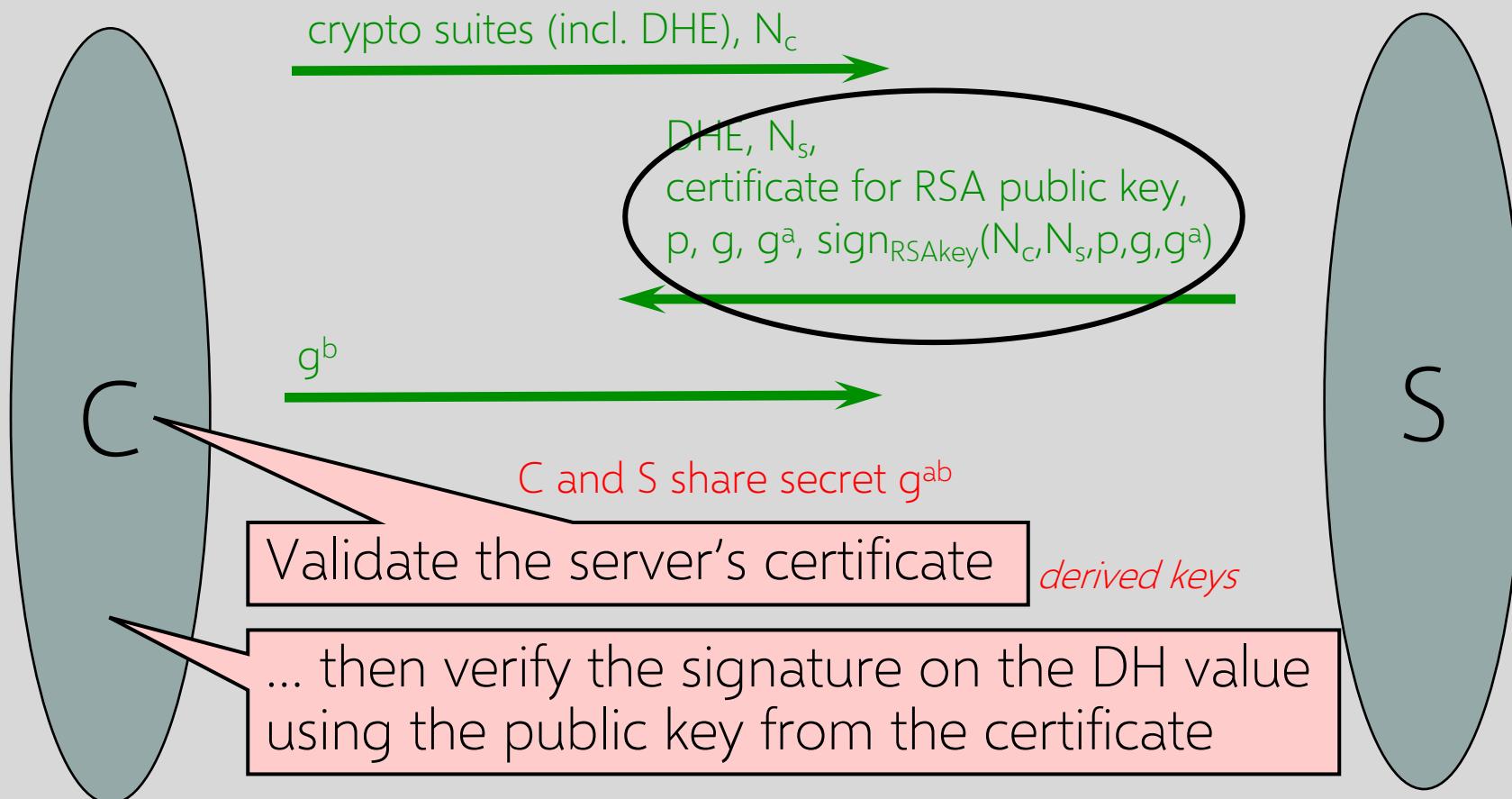
A Perfect CRIME

Only TIME Will

Tell

Padding oracles
Compression oracles
Downgrades to export cryptography
... many other attacks over the years

More Fun with Diffie-Hellman



MITM Presenting a Valid Certificate



Hello

I am PayPal.com (or whoever you want me to be)
Here is PayPal's certificate for its RSA signing key
And here is my signed Diffie-Hellman value



Validate the server's certificate

... then verify the signature on the DH value
using the public key from the certificate

Goto Fail



Here is PayPal's certificate
And here is my signed Diffie-Hellman value



... verify the signature on the DH value using
the public key from the certificate

```
if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail; ??? ←
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail; ...
err = sslRawVerify(...); ← Signature is verified here
...
fail: ... return err ...
```

Complete Fail Against MITM

Discovered in February 2014

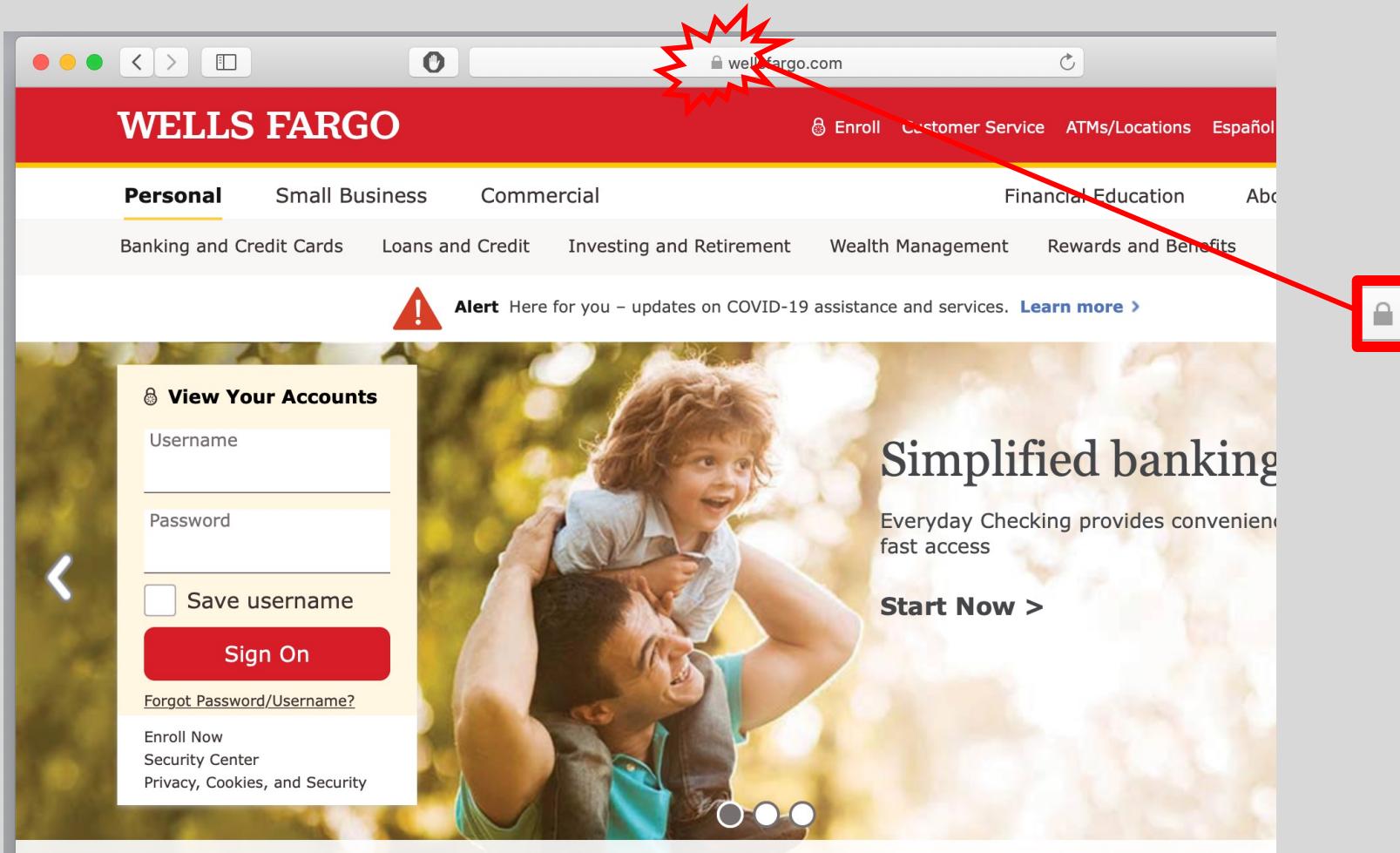
All OS X and iOS software vulnerable to man-in-the-middle attacks

- Broken TLS implementation provides no protection against the very attack it was supposed to prevent

What does this tell you about quality control for security-critical software?



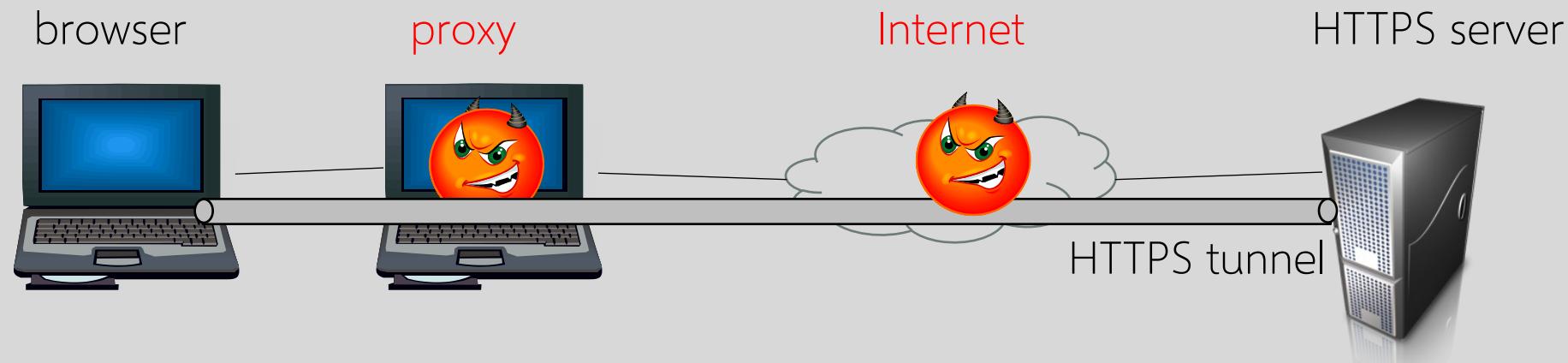
HTTPS: HTTP over SSL/TLS



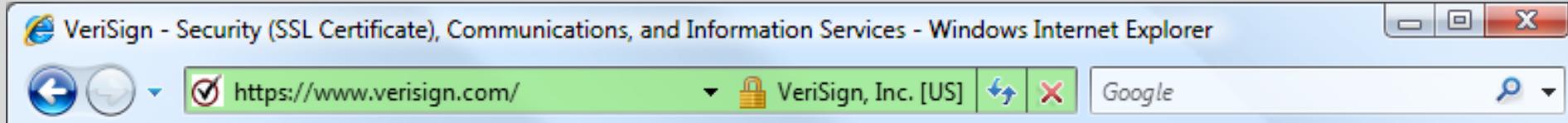
HTTPS and Its Adversary Model

HTTPS: **end-to-end** secure protocol for Web

Goals: confidentiality, authentication (usually for server only), and integrity against network attackers, including man-in-the-middle (MITM) attacks



The Lock Icon



Goal: identify secure connection

- SSL/TLS is being used between client and server to protect against active network attacker

Lock icon should only be shown when the page is secure against network attacker

- Semantics subtle and not widely understood by users
- Problem in user interface design

HTTPS Security Guarantees



The origin of the page is what it says in the address bar

Contents of the page have not been viewed or modified
by a network attacker

User must interpret what he sees

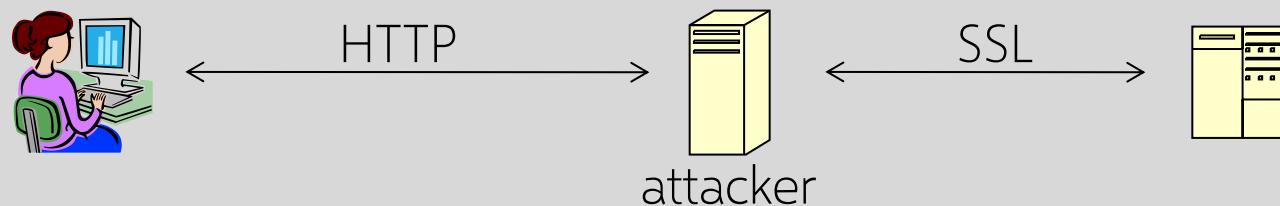
HTTP → HTTPS and Back

Can the server
detect this attack?

Common pattern: HTTPS upgrade

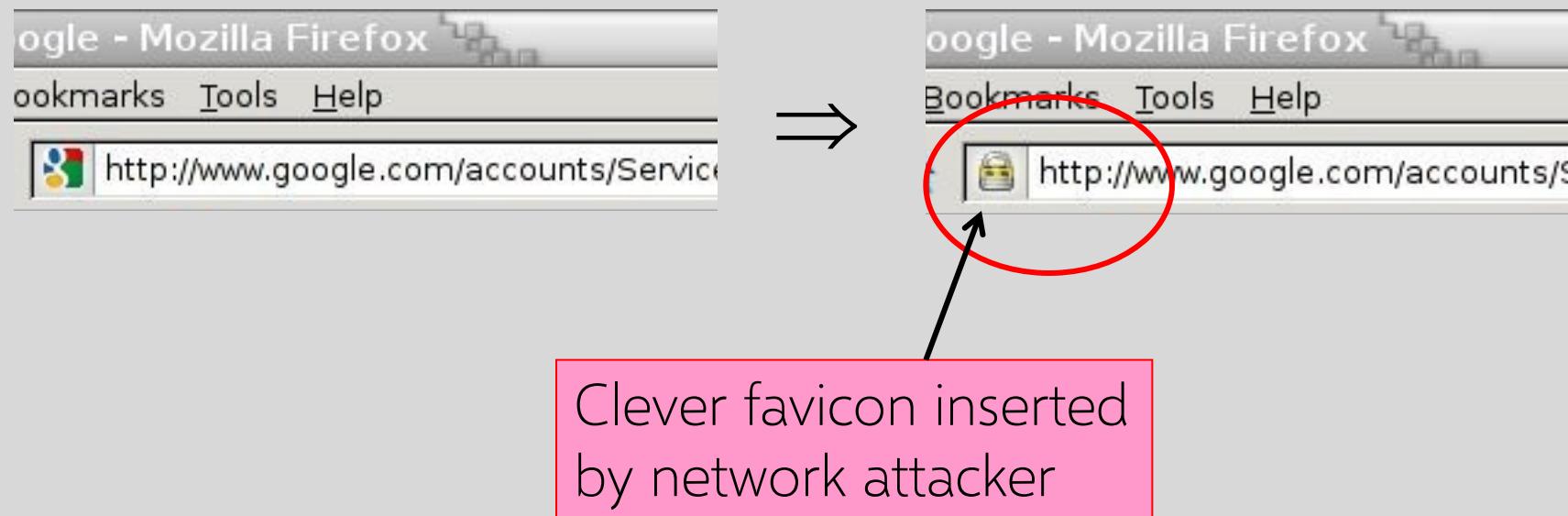
- Come to site over HTTP, redirect to HTTPS for login
- Browse site over HTTP, redirect to HTTPS for checkout

sslstrip: man-in-the-middle network attacker downgrades connection



Rewrite to
Redirect Location: https://... to Location: http://...
Rewrite <form action="https://...> to <form action="http://...>

Will You Notice?



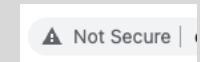
Browsers no longer put favicons into the address bar

Source: Moxie Marlinspike

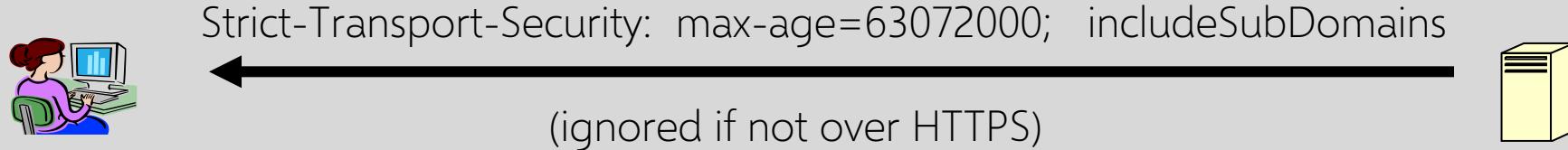
Encrypt all the things!
HTTPS EVERYWHERE!!!



Since 2018, Chrome marks
all HTTP sites as insecure



HSTS: Strict Transport Security



Header tells browser to always connect over HTTPS

- Subsequent visits must be over HTTPS
- Browser refuses to connect over HTTP or if site presents an invalid or self-signed cert

Requires that entire site be served over valid HTTPS

HSTS flag deleted when user “clears private data”: security vs. privacy

Preloaded HSTS List

Enter a domain for the HSTS preload list:

paypal.com

Check status and eligibility

Strict-Transport-Security: max-age=63072000; includeSubDomains; preload

Preload list hard-coded in Chrome source code.

Examples: Google, Paypal, Twitter, Simple, Linode, Stripe, Lastpass, ...

Using CSP to Upgrade to HTTPS

Problem: many pages use ``

Makes it difficult to migrate a section of a site to HTTPS

Solution: gradual transition using CSP

Content-Security-Policy: upgrade-insecure-requests

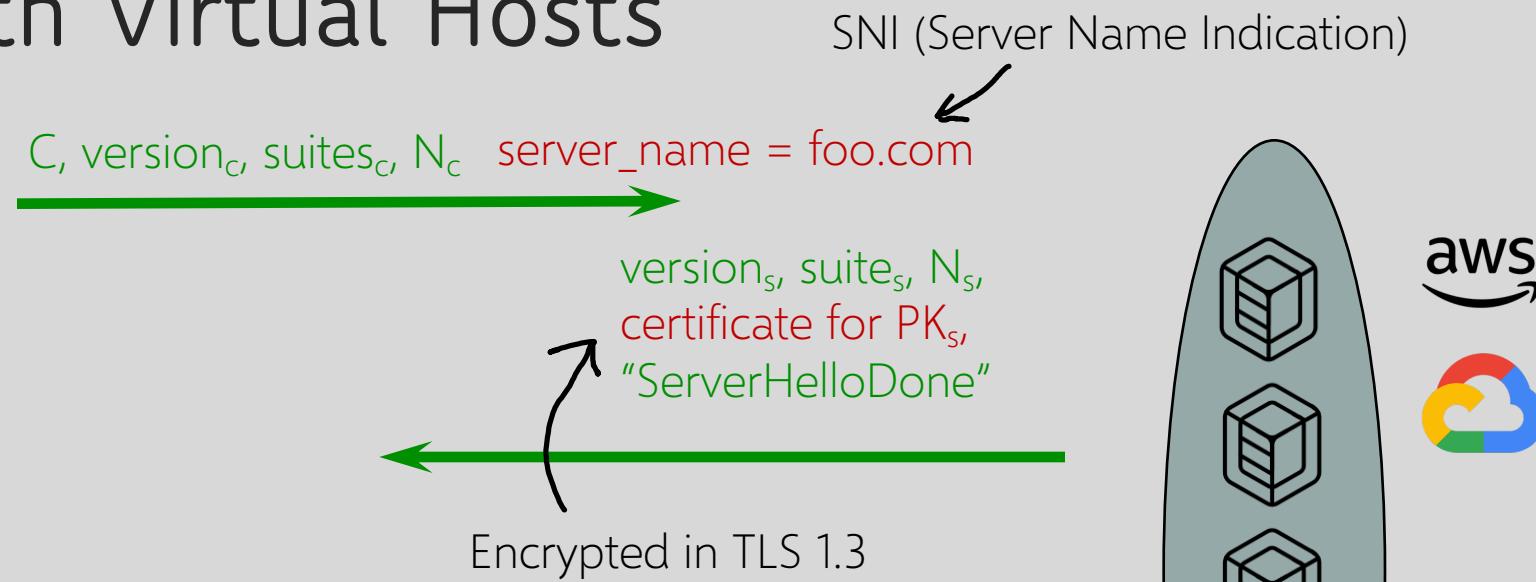
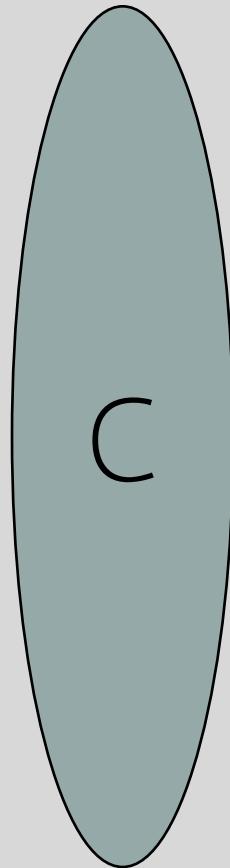
`` ``

`` → ``

`` ``

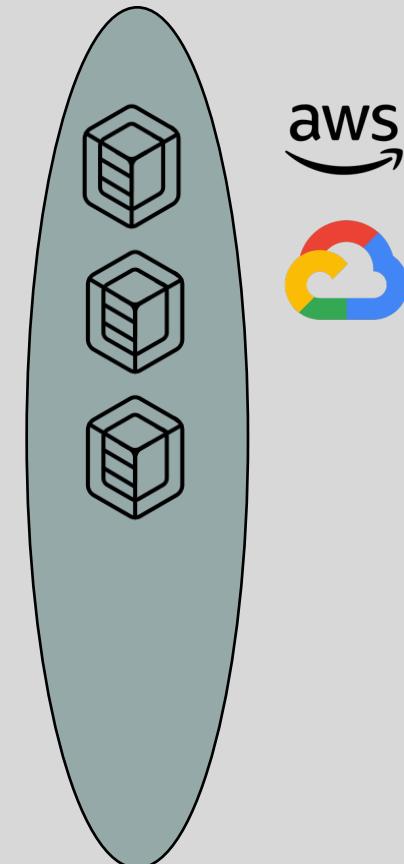
`` ``

Dealing with Virtual Hosts

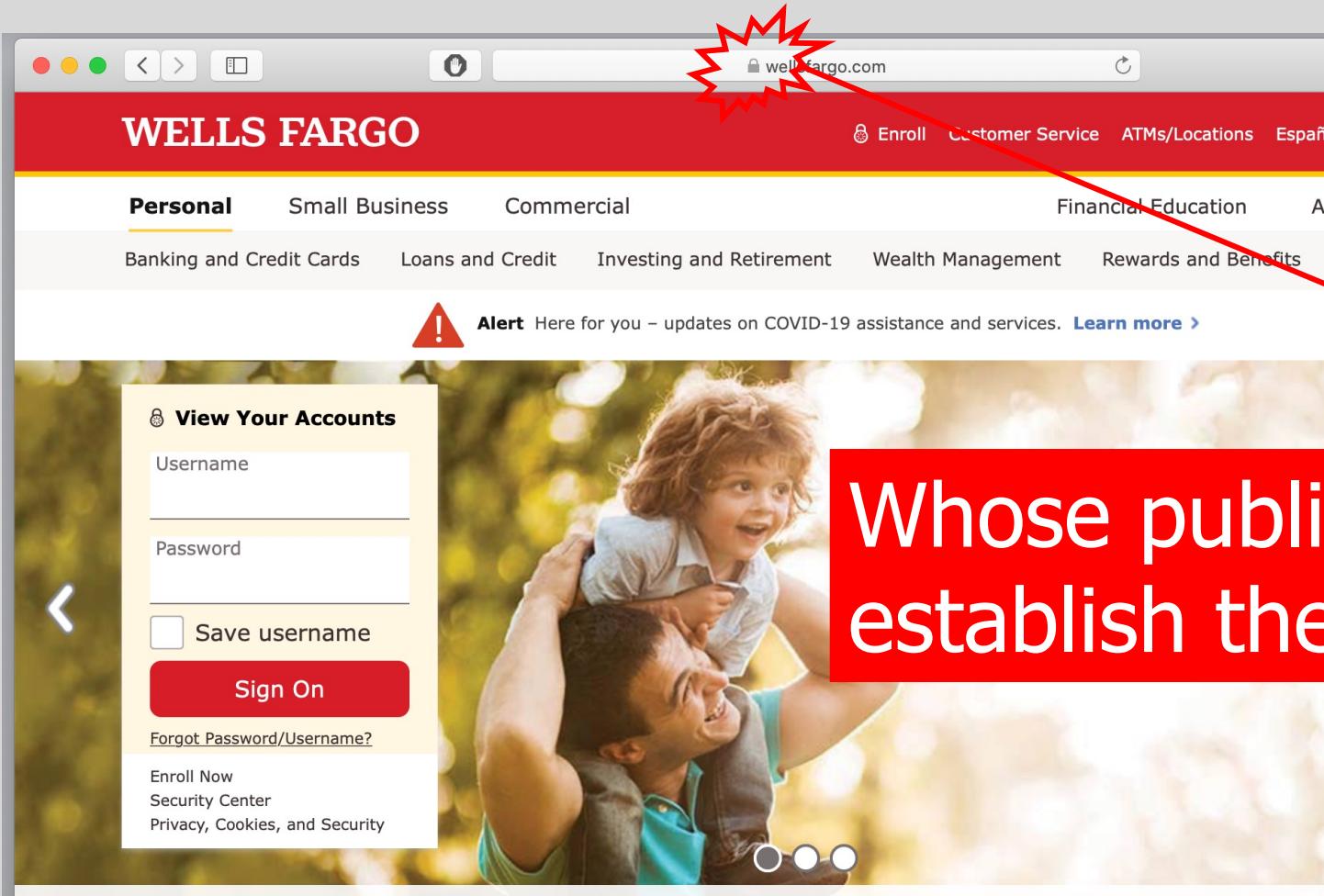


How to hide which server is
accessed from eavesdroppers?

Solution: encrypt the entire ClientHello message
... with what key?
Include public key in the server's DNS record

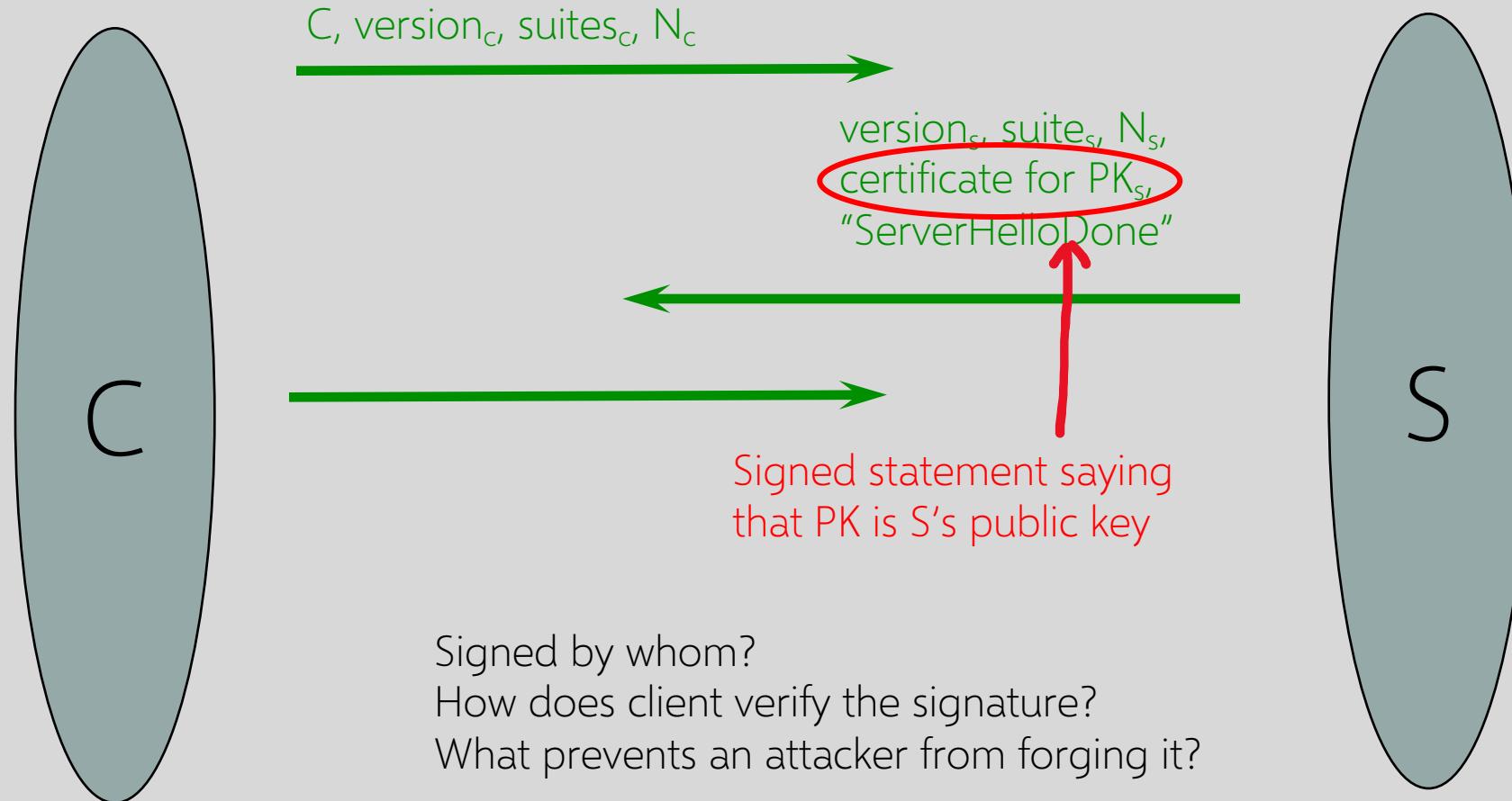


Authenticating the Server



Whose public key is used to establish the secure session?

Public-Key Certificates



Certificate Authorities

A **public-key certificate** is a signed statement specifying the key and identity

- Could be signed by the subject itself (“self-signed”), but why would anyone trust such a certificate?

Instead: **certificate authorities (CAs)**

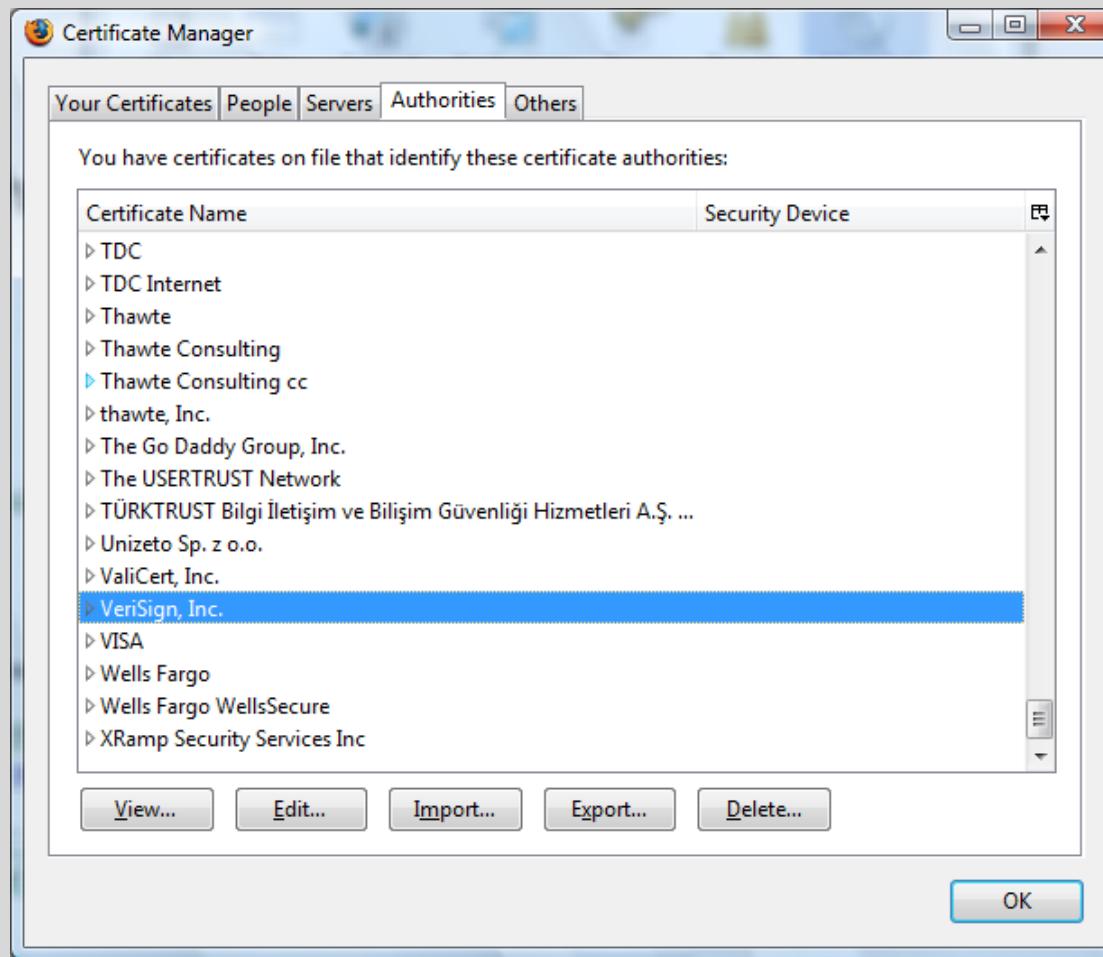
- Agencies responsible for certifying public keys
- Browsers are pre-configured with the public keys of 100+ of trusted CAs
- A public key for any website in the world will be accepted by the browser if the certificate is signed by one of these CAs



CA hold “skeleton keys” to the entire Internet



Trusted Certificate Authorities



Example of a Certificate

Certificate Viewer: "5654961308303360-fe2.pantheonsite.io"

General Details

This certificate has been verified for the following uses:

SSL Server Certificate

Issued To

Common Name (CN)	5654961308303360-fe2.pantheonsite.io
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	03:50:CF:80:74:39:79:89:70:4F:D0:94:00:D4:42:50:91:EA

Issued By

Common Name (CN)	Let's Encrypt Authority X3
Organization (O)	Let's Encrypt
Organizational Unit (OU)	<Not Part Of Certificate>

Period of Validity

Begins On	October 26, 2019
Expires On	January 24, 2020

Fingerprints

SHA-256 Fingerprint	56:46:FE:46:64:42:77:F6:8E:78:0B:9E:C8:D3:5F:C4: 9C:9C:27:8F:A5:78:0F:C7:E1:15:10:58:4D:5B:42:26
SHA1 Fingerprint	23:85:1B:04:D2:76:88:18:EC:0D:1D:D3:A1:E7:C0:09:5F:99:0F:65

Certificate Signature Algorithm

Issuer

Validity

Not Before

Not After

Subject

Subject Public Key Info

Subject Public Key Algorithm

Subject's Public Key

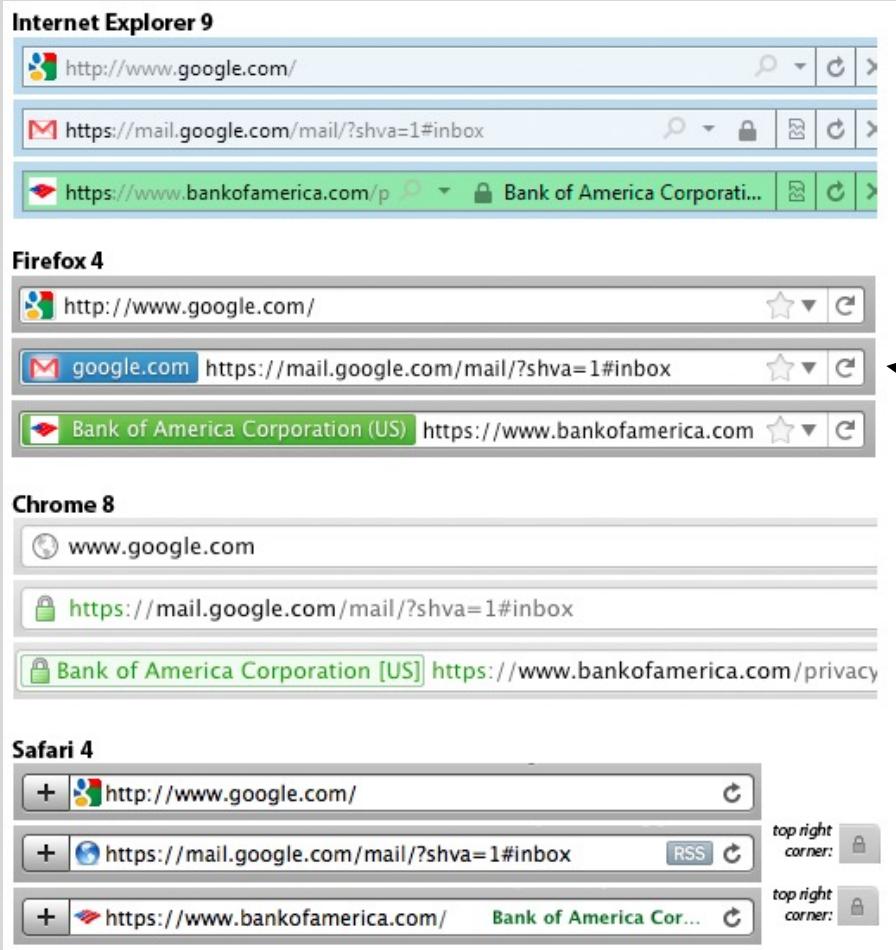
Extensions

Field Value

Modulus (1024 bits):

```
ac 73 14 97 b4 10 a3 1 f4 c1 15 ed cf 92 f3 9a  
97 26 9a cf 1b e4 1b dc d2 c9 37 2f d2 e6 07 1d  
ad b2 3e f7 8c 2f fa a1 b7 9e e3 54 40 34 3f b9  
e2 1c 12 8a 30 6b 0c fa 30 6a 01 61 e9 7c b1 98  
2d 0d c6 38 03 b4 55 33 7f 10 40 45 c5 c3 e4 d6  
6b 9c 0d d0 8e 4f 39 0d 2b d2 e9 88 cb 2d 21 a3  
f1 84 61 3c 3a aa 80 18 27 e6 7e f7 b8 6a 0a 75  
e1 bb 14 72 95 cb 64 78 06 84 81 eb 7b 07 8d 49
```

The Meaning of Color



What is the difference?

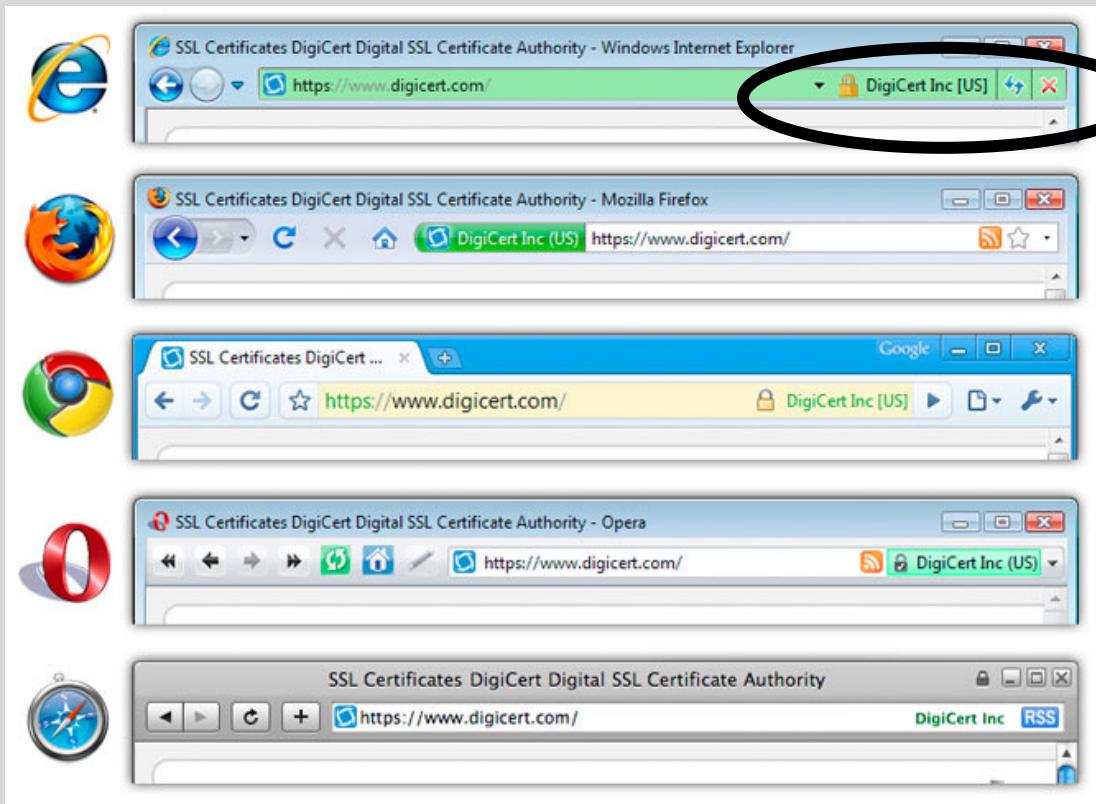
Domain Validation (DV) certificate
vs.
Extended Validation (EV) certificate

Means what?

Source: Schulze

Extended Validation (EV) Certificates

EV certificate request must be approved by a human at the certificate authority



Helps block “semantic attacks”:
Remember www.bankofthevest.com?

UI ineffective, removed from Chrome in 2019

Questions About EV Certificates

What does EV certificate mean?

What is the difference between an HTTPS connection that uses a regular certificate and an HTTPS connection that uses an EV certificate?

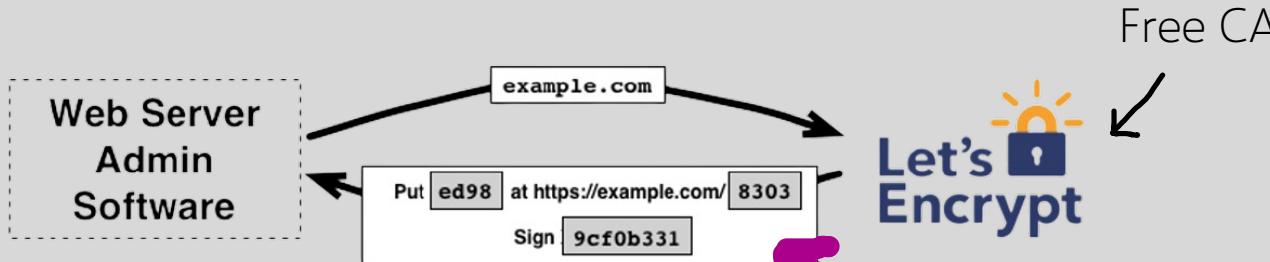
If an attacker has somehow obtained a non-EV certificate for bank.com, can he inject a script into <https://bank.com> content?

- What is the origin of the script? Can it access or modify content that arrived from actual bank.com via HTTPS?

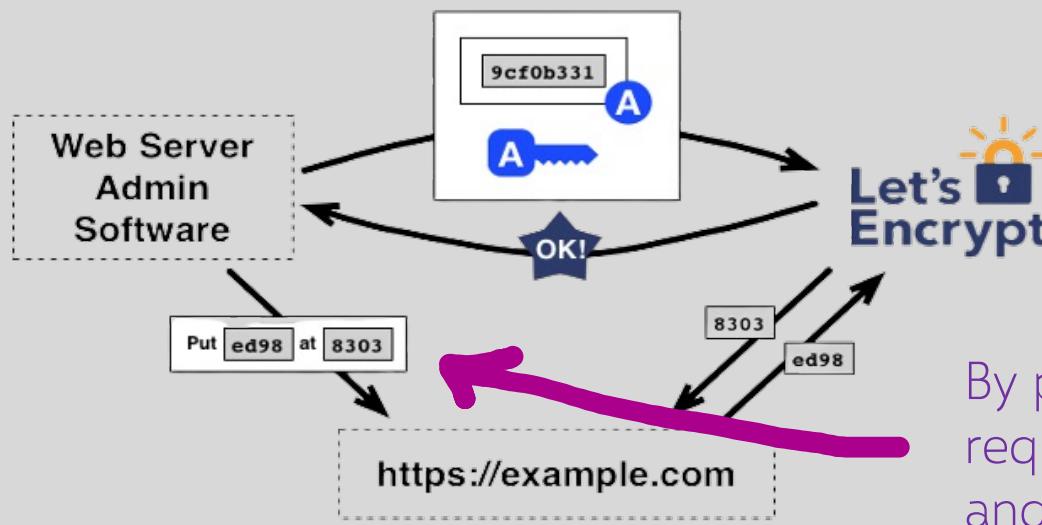
What would the browser show – blue or green?

Certificate Issuance

How does this work with wildcard certificates like *.cornell.edu?



CA gives a challenge to the requester



By putting a signed challenge on the website, requester proves that he controls the website and knows the signing key (why?)

CA Hierarchy

Browsers, operating systems, etc. have trusted root certificate authorities

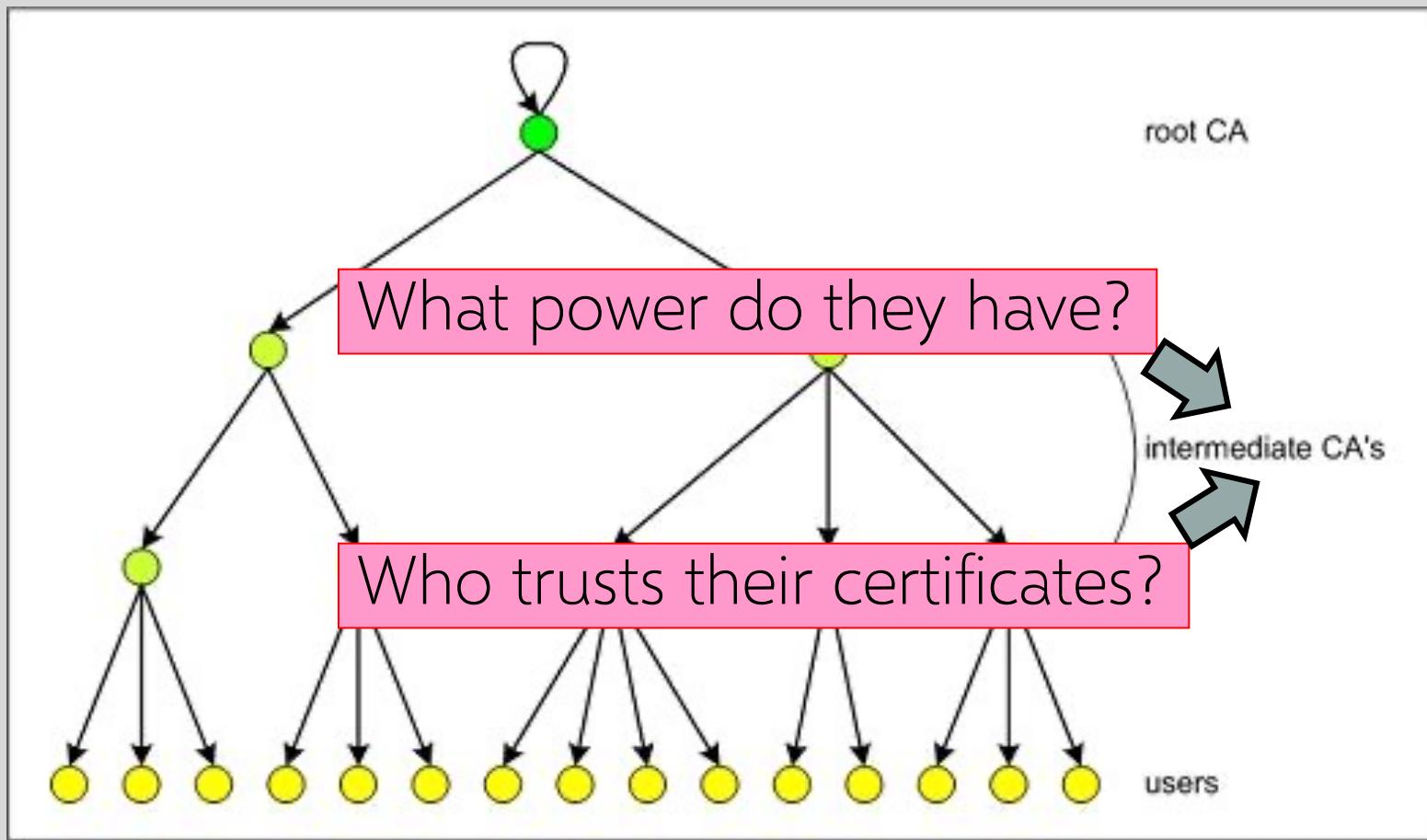
- Chrome includes certificates of ~200 trusted root CAs

Root CA signs certificates for intermediate CAs, they sign certificates for lower-level CAs, etc.

- “Chain of trust”: $\text{sig}_{\text{Verisign}}(\text{"Cornell"}, \text{PK}_{\text{Cornell}})$,
 $\text{sig}_{\text{Cornell}}(\text{"Vitaly S."}, \text{PK}_{\text{Vitaly}})$

CAs are responsible for verifying the identities of certificate requestors, domain ownership

Certificate Hierarchy



How Many?

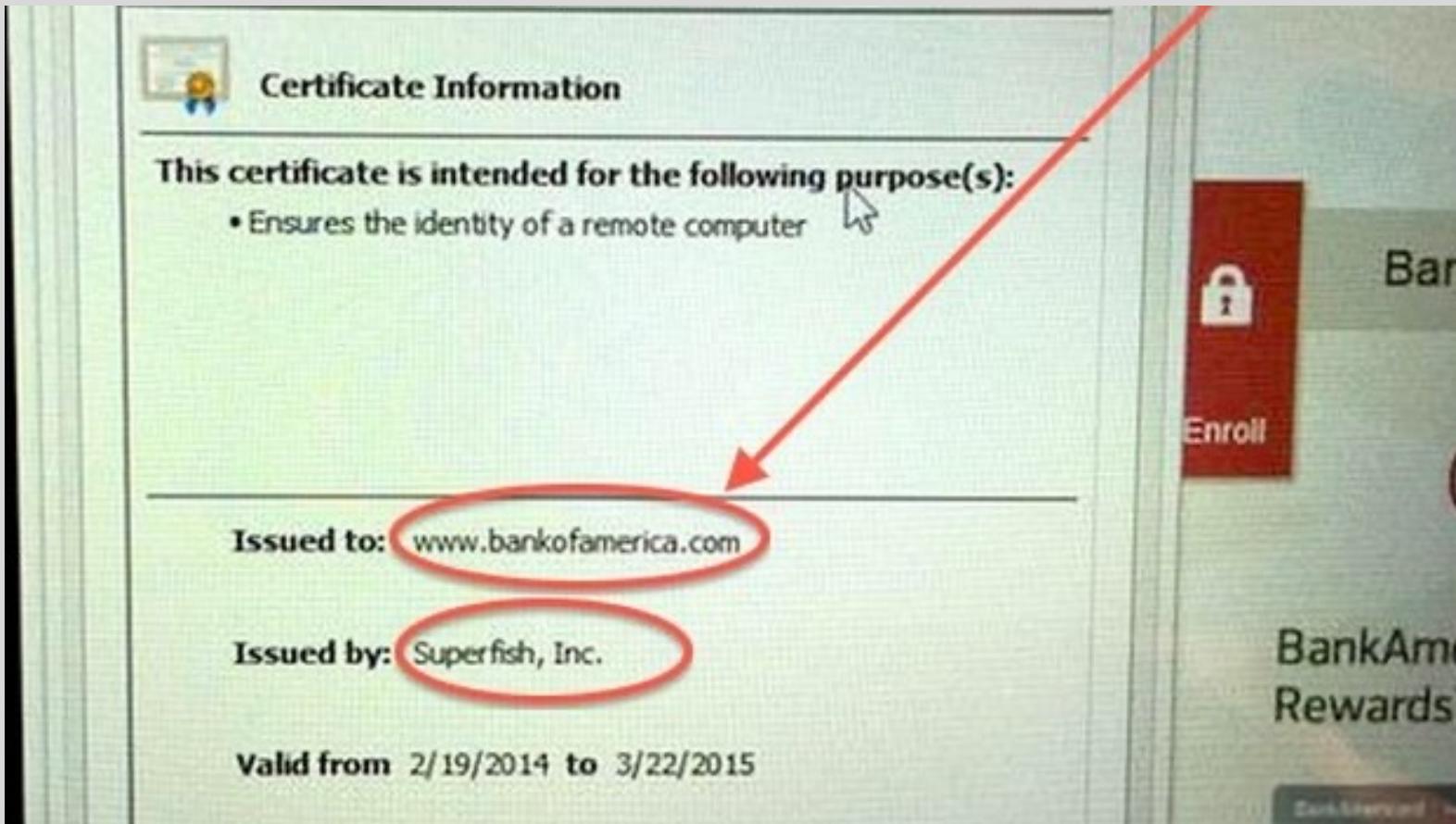
"1,800 entities that are able to issue certificates vouching for the identity of any website"

Durumeric et al. Analysis of the HTTPS Certificate Ecosystem (2013)

A public key for any website in the world will be accepted by the browser if the certificate is signed by one of these CAs



Another Example of a Certificate



Root Certificates in Lenovo Laptops (2015)

In the news



[Lenovo hit by lawsuit over Superfish adware](#)

CNET - 3 days ago

Sarah Tew/CNET. **Lenovo** may find itself in a courtroom over its **Superfish adware** fiasco.

[Interview with Lenovo's CTO will scare anyone still thinking of buying a Lenovo product](#)
BGR - 2 days ago

[Lenovo's Chief Technology Officer Discusses the Superfish Adware Fiasco - NYTimes.com](#)

Bits - The New York Times - 3 days ago

[More news for lenovo superfish adware](#)

[Lenovo Sued Over Superfish Adware : NPR](#)

[www.npr.org](#) › News › Business

NPR ▾

2 days ago - Renee Montagne talks to Jordan Robertson of Bloomberg News about computer maker **Lenovo**, which allowed controversial spyware to be ...

[Lenovo users lawyer up over hole-filled, HTTPS-breaking ...](#)

[arstechnica.com/.../lenovo-users-lawyer-up-over-hole-filled...](#) ▾ Ars Technica ▾

4 days ago - In the wake of last week's **Lenovo's Superfish** debacle, at least one ... and that **Superfish adware** "does not present a security risk," despite ...

[Lenovo's Chief Technology Officer Discusses the Superfish ...](#)

[bits.blogs.nytimes.com/.../lenovos-chief-technology-officer-discusses-the...](#) ▾

3 days ago - The **adware** was intended to serve **Lenovo** users targeted ads, but the company **Lenovo** partnered with to do this, **Superfish**, did so by hijacking ...

[Lenovo Sued Over Superfish Adware | News & Opinion ...](#)

[www.pcmag.com](#) › Reviews › Software › Security

PC Magazine ▾

4 days ago - Not surprisingly, the controversy over **Lenovo** installing **Superfish adware** into its consumer PCs has resulted in a lawsuit. According to the suit, ...

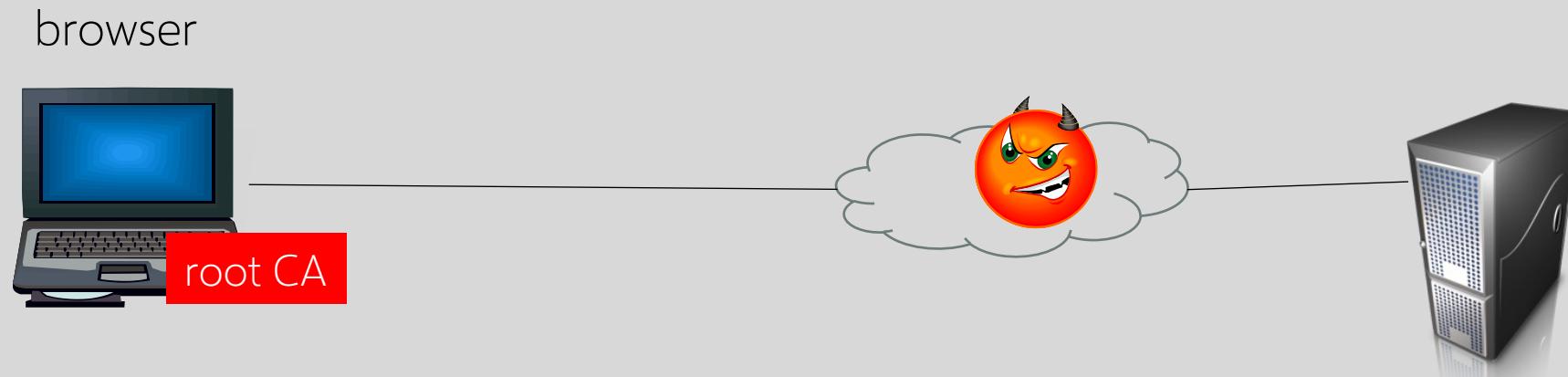
What Could You Do ...

... if you had a certificate for cornell.edu
signed by a root CA?

Man-in-the-Middle (MITM) Attack

Read all traffic, undetected

Impersonate any website, undetected



Flame

Flame: The Most Sophisticated
Cyber Espionage Tool Ever Made

Cyber-espionage virus (2010-2012)

Signed with a fake intermediate CA
certificate accepted by any Windows
Update service

- Fake certificate was created using an MD5 chosen-prefix collision against an obscure Microsoft Terminal Server Licensing Service certificate that was enabled for code signing

MD5 collision technique possibly pre-dates
academic publication of MD5 collisions

- Evidence of state-level cryptanalysis?



Comodo is one of the trusted root CAs

- Its certificates for any website in the world are accepted by every browser

Comodo accepts certificate orders submitted through resellers

- Reseller uses a program to authenticate to Comodo and submit an order with a domain name and public key, Comodo automatically issues a certificate for this site

Comodo Break-In

What are potential consequences?

An Iranian hacker broke into instantSSL.it and globalTrust.it resellers, decompiled their certificate issuance program, learned the credentials of their reseller account and how to use Comodo API

- username: gtadmin, password: globaltrust

Wrote his own program for submitting orders and obtaining Comodo certificates

On March 15, 2011, got Comodo to issue 9 rogue certificates for popular sites

- mail.google.com, login.live.com, login.yahoo.com, login.skype.com, addons.mozilla.org, “global trustee”

Message from the Attacker

I'm single hacker with experience of 1000 hacker, I'm single programmer with experience of 1000 programmer, I'm single planner/project manager with experience of 1000 project managers ...

When USA and Isarel could read my emails in Yahoo, Hotmail, Skype, Gmail, etc. without any simple little problem, when they can spy using Echelon, I can do anything I can. It's a simple rule. You do, I do, that's all. You stop, I stop. It's rule #1 ...

Rule#2: So why all the world got worried, internet shocked and all writers write about it, but nobody writes about Stuxnet anymore?... So nobody should write about SSL certificates.

Rule#3: I won't let anyone inside Iran, harm people of Iran, harm my country's Nuclear Scientists, harm my Leader (which nobody can), harm my President, as I live, you won't be able to do so. as I live, you don't have privacy in internet, you don't have security in digital world, just wait and see...

An update on attempted man-in-the-middle attacks

August 29, 2011

Posted by Heather Adkins, Information Security Manager

Today we received reports of attempted SSL man-in-the-middle (MITM) attacks against Google users, whereby someone tried to get between them and encrypted Google services. The people affected were primarily located in Iran. The attacker used a fraudulent SSL certificate issued by DigiNotar, a root certificate authority that should not issue certificates for Google (and has since revoked it).

DigiNotar Break-In

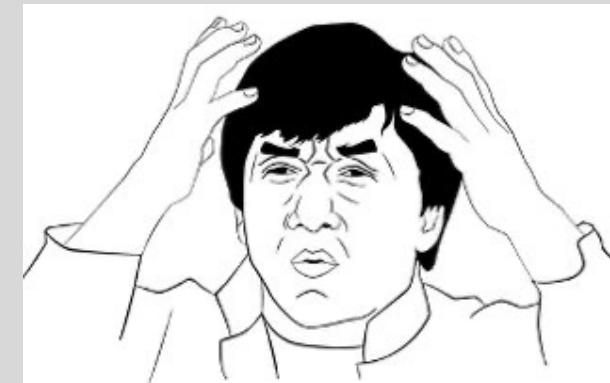
In June 2011, “ComodoHacker” broke into a Dutch certificate authority, DigiNotar



Message found in scripts used to generate fake certificates: **“THERE IS NO ANY HARDWARE OR SOFTWARE IN THIS WORLD EXISTS WHICH COULD STOP MY HEAVY ATTACKS MY BRAIN OR MY SKILLS OR MY WILL OR MY EXPERTISE”**

Security of DigiNotar servers:

- All core certificate servers in a single Windows domain, controlled by a single admin password (Pr0d@dm1n)
- Software on public-facing servers out of date, unpatched
- Tools used in the attack would have been easily detected by an antivirus... if it had been present



Consequences of the DigiNotar Hack

Break-in not detected for a month

Rogue certificates issued for *.google.com, Skype, Facebook, www.cia.gov, and 527 other domains

99% of revocation lookups for these certificates originated from Iran

- Evidence that rogue certificates were being used, most likely by Iranian government or Iranian ISPs to intercept encrypted communications
- 300,000 users were served rogue certificates (95% in Iran)

Textbook
“man in the
middle”



Another Message from the Attacker

Most sophisticated hack of all time ... I'm really sharp, powerful, dangerous and smart!

My country should have control over Google, Skype, Yahoo, etc. [...] I'm breaking all encryption algorithms and giving power to my country to control all of them.

You only heards Comodo (successfully issued 9 certs for me -thanks by the way-), DigiNotar (successfully generated 500+ code signing and SSL certs for me -thanks again-), StartCOM (got connection to HSM, was generating for twitter, google, etc. CEO was lucky enough, but I have ALL emails, database backups, customer data which I'll publish all via cryptome in near future), GlobalSign (I have access to their entire server, got DB backups, their linux / tar gzipped and downloaded, I even have private key of their OWN globalsign.com domain, hahahaa).... BUT YOU HAVE TO HEAR SO MUCH MORE! SO MUCH MORE! At least 3 more, AT LEAST

Rogue Certificates in the Wild



In Jan 2013, TurkTrust (a root CA) accidentally issued intermediate CA certificates to customers who requested regular certificates

Ankara transit authority used its certificate to issue a fake *.google.com certificate in order to intercept and filter SSL traffic from its network

More rogue certs (2015)

- MCS Holdings (Egypt) for Google domains
 - Root CA: CNNIC (China)
- WoSign (Chinese CA) for Github and Alibaba

These guys gained the ability to intercept HTTPS traffic to any website in the world



Rogue Certs for Surveillance

- Google, Mozilla, and Apple have blocked an encryption certificate issued by the Kazakhstan government, which citizens were asked to install on their browsers and that critics said enabled the government to monitor their internet traffic.
- The government reportedly said the software was a security measure but researchers from the University of Michigan found that installing the browser certificate allowed the government to surveil which sites people were accessing, and see anything a user types or posts.
- According to the researchers, the fake certificate targeted 37 sites including Google-owned messaging apps, Google Docs, Instagram, Gmail, Twitter, Facebook, and a number of Russian social media services.



<https://www.businessinsider.com/apple-google-mozilla-block-kazakhstan-governments-browser-spying-tool-2019-8>



In Feb 2012, TrustWave admitted issuing an intermediate CA certificate to a corporate customer

- Purpose: “re-sign” certificates for “data loss prevention”
- Translation: forge certificates of third-party sites in order to spy on employees’ encrypted communications with the outside world

Customer can now forge certificates for any site in world...
and they will be accepted by any browser!

- What if a “re-signed” certificate leaks out?



Software behind the
Superfish / Lenovo scandal

Defunct Israeli startup

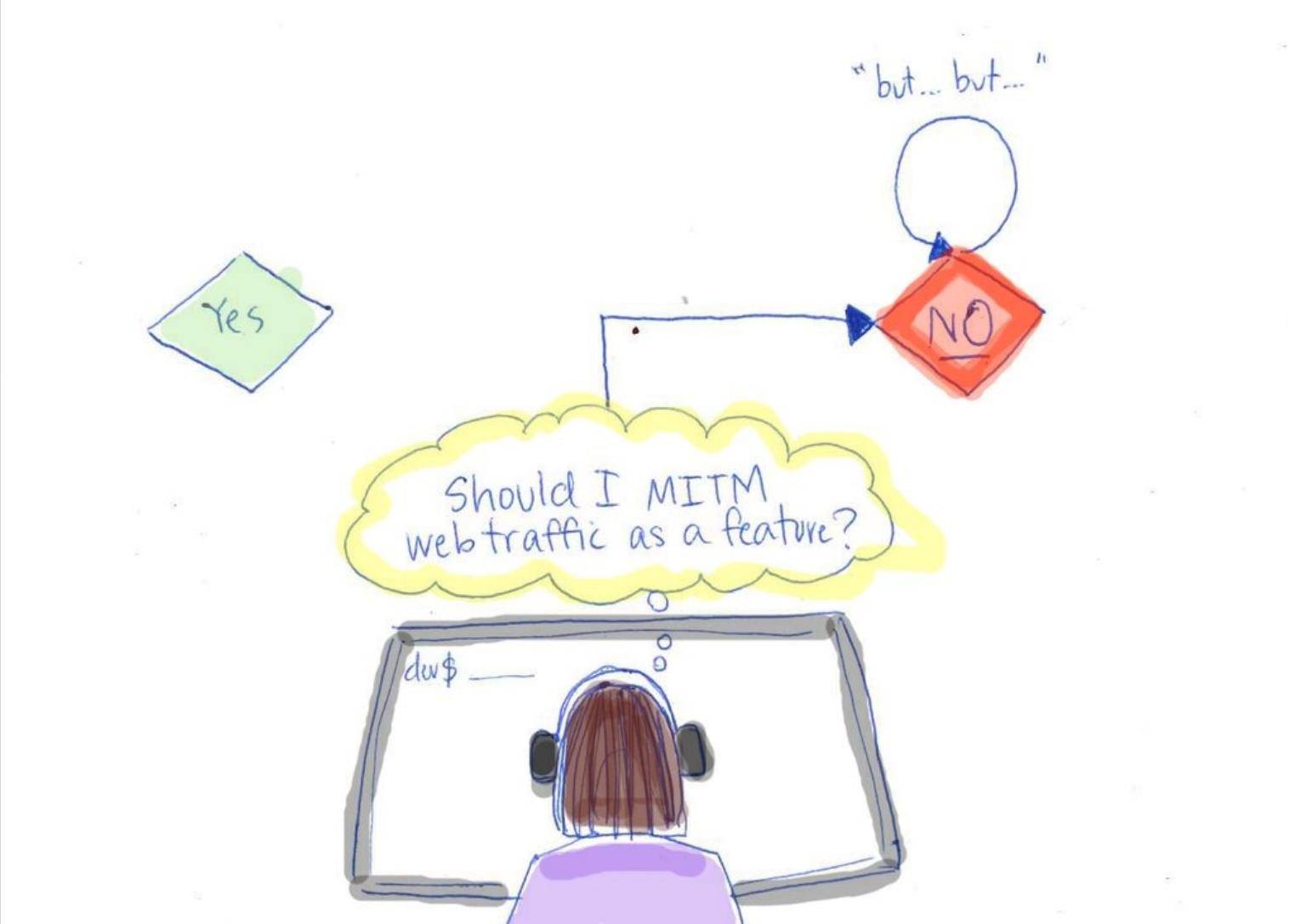
Installed its own root certificate

- Goal: re-sign SSL certificates, proxy/MITM connections
- From their former website: “Our **advanced SSL hijacker SDK** is a brand new technology that allows you to access data that was encrypted using SSL and perform on the fly SSL decryption.”

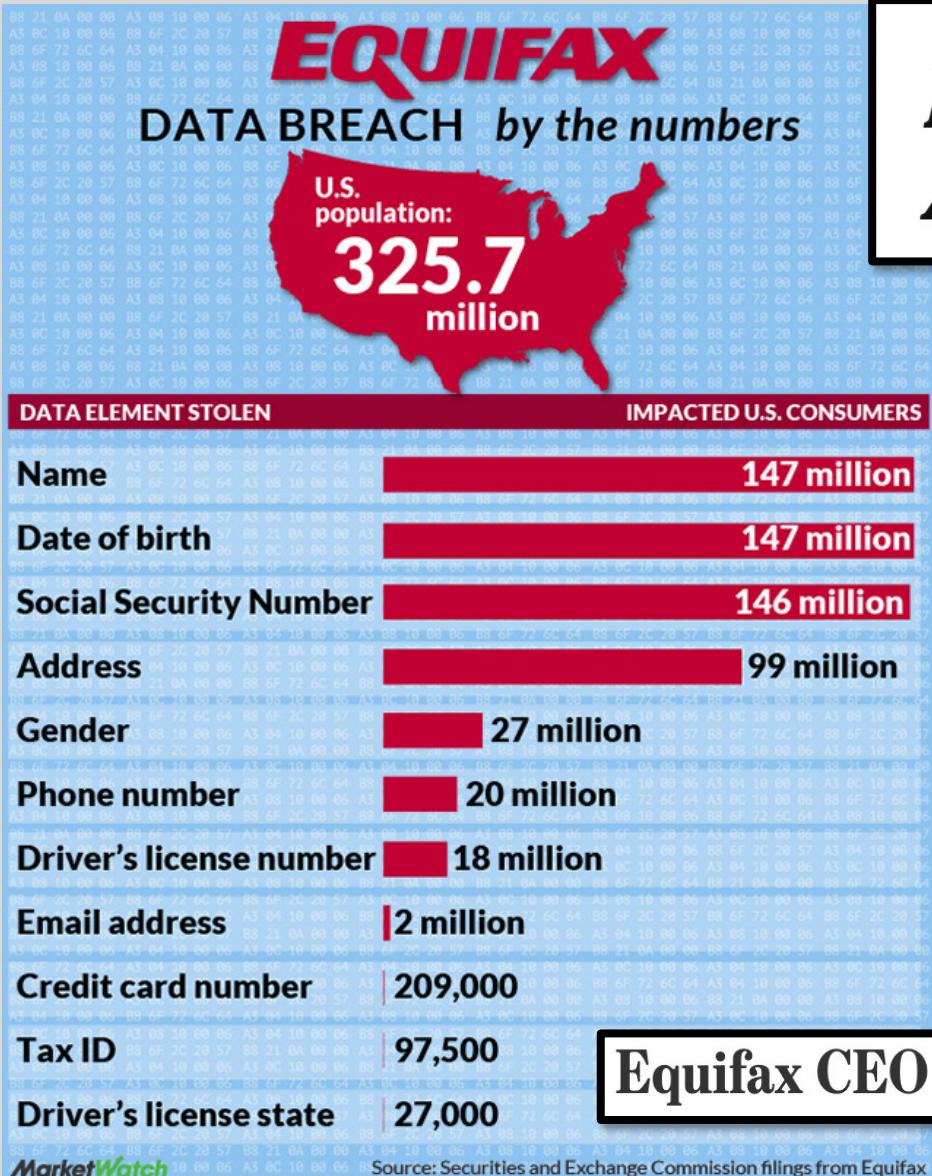
Same private key on all machines, easily extracted

- Anyone could issue fake Komodia certificates, MITM any machine with Komodia

Also turned self-signed certificates into trusted certificates



Credit: Adrienne Porter Felt (Google)



Equifax Says Cyberattack May Have Affected 143 Million in the U.S.

Equifailure

Share prices, January 1st 2017=100, \$ terms



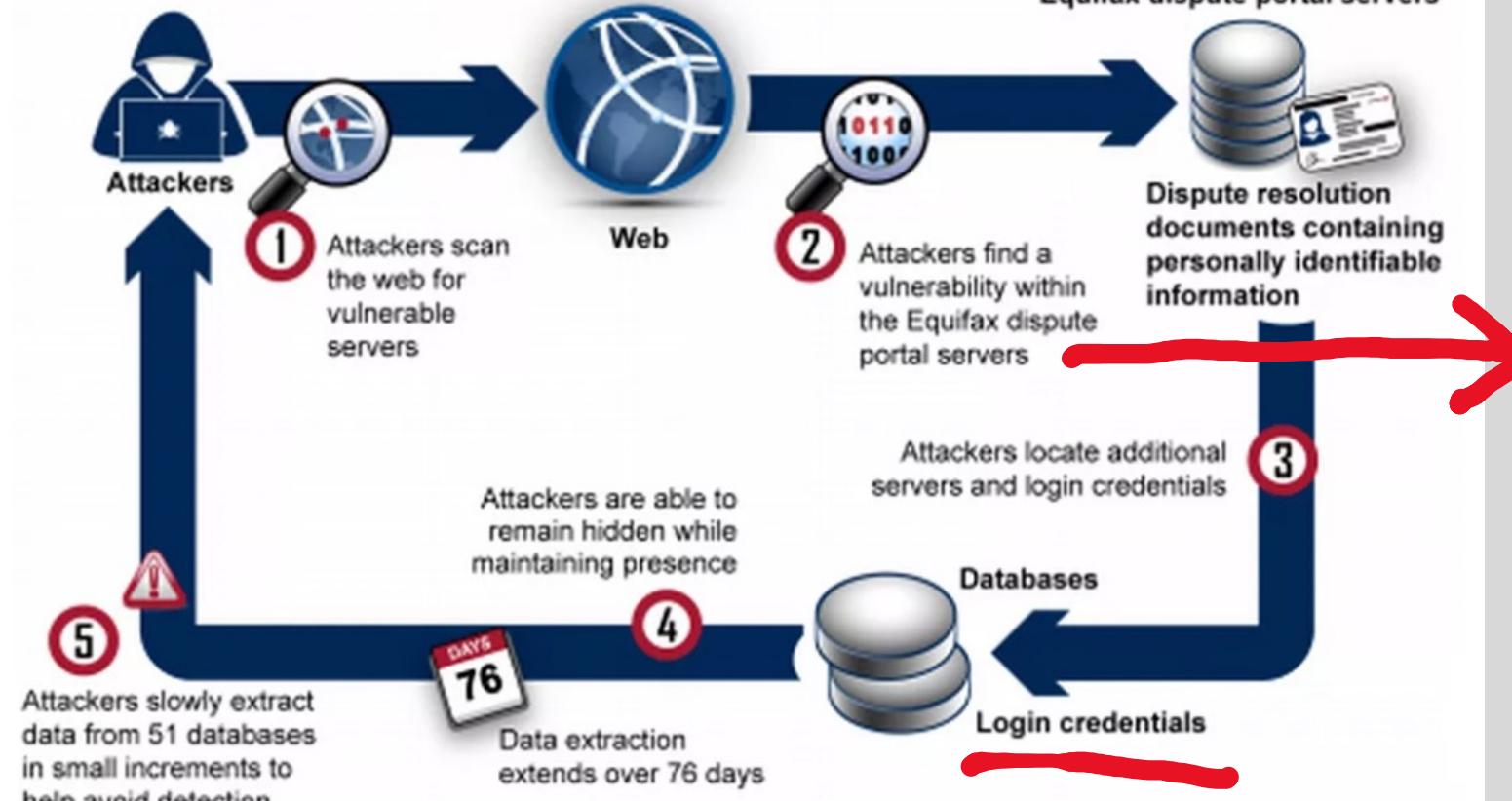
Source: Thomson Reuters

Economist.com

Former CIO of a business unit dumped \$1 million in stock, later charged with insider trading

Equifax CEO Richard Smith to Exit Following Massive Data Breach

How Attackers Exploited Vulnerabilities in the 2017 Breach, Based on Equifax Information



Source: GAO, based on information provided by Equifax. | GAO-18-559

United States Government Accountability Office

CVE-2017-5638
vulnerability in Apache
Struts, an open-source
framework for enterprise
Java applications



If code is included in the Content-Type header of an HTTP request, file upload parser tricked into executing this code

Timeline of the Equifax Hack

- Early March, 2017: Struts vulnerability discovered
- March 7: Apache released a patch
- March 9: Equifax IT told to apply the patch to all affected systems... they didn't
- March 10: Equifax Web portal breached via the Struts vulnerability
- March 15: Equifax IT ran a series of scans to identify unpatched systems... the scan did not identify any of vulnerable, unpatched systems
- ...
- May 13: Attackers start moving from compromised server into other parts of the network and exfiltrating data

<https://www.csoonline.com/article/3444488/equifax-data-breach-faq-what-happened-who-was-affected-what-was-the-impact.html>

Detecting Data Exfiltration

Attackers were encrypting the data they were stealing from Equifax

Like many enterprises, Equifax decrypts, inspects, and re-encrypts all internal network traffic

The SSL certificate needed for re-encryption expired 10 months prior... exfiltration was discovered only on July 29, 2017, when the certificate was renewed

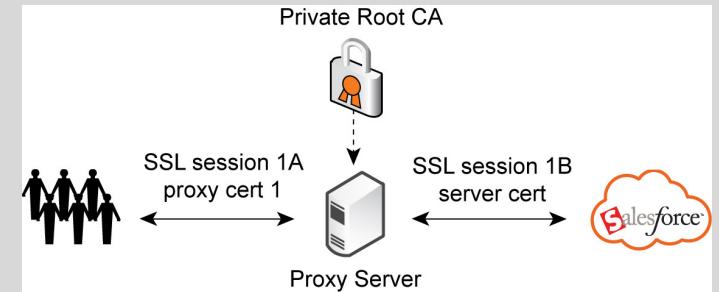
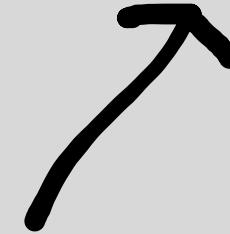


Image: Riverbed



Russian hackers modify Chrome and Firefox to track secure web traffic

Malware installs its own certificate

Modifies pseudo-random generator to add encrypted hardware and software identifiers of the infected machine to the random nonce sent as part of the TLS handshake

<https://securelist.com/comfun-successor-reductor/93633/>

Revoking Certificates

Short expirations

CRLs (certificate revocation lists)

OCSP (online certificate status protocol)

- Client queries CA to check validity of cert
 - Privacy concerns, performance / scalability issues
- Stapling: website periodically gets fresh, time-stamped OCSP signature from CA, includes it with the certificate
 - Certs can have “must staple” extension

Revoking DigiNotar Certificates

The discovery of the DigiNotar compromise left the browser and CA community—to say nothing of the Dutch government—reeling. Browser vendors rushed to revoke trust in DigiNotar certificates, but removing a root CA was not entirely straightforward. “We actually needed to push out an update to Firefox because the CA information was hard-coded to the browser,” Firefox security lead Richard Barnes said. Additionally, many legitimate websites (including some operated by the Dutch government) were still relying on DigiNotar certificates, so the browser vendors were forced to hold off on a blanket ban. Instead, Mozilla decided to block all DigiNotar certificates issued after July 1, 2011, but allowed users to decide whether they wanted to trust certificates issued by the company before that date. But giving users that

In the Netherlands, interior minister went on TV to warn Dutch citizens to immediately stop using secure government websites

March 13, 2019

3

Mass Revocation: Millions of certificates revoked by Apple, Google & GoDaddy

The DarkMatter debate is already having industry-wide ramifications

Millions of SSL/TLS certificates – among other digital certificates – are being revoked right now as a result of an operational error that caused the generation of non-compliant serial numbers.

<https://www.thesslstore.com/blog/mass-revocation-millions-of-certificates-revoked-by-apple-google-godaddy/>

Domain owners use CAA field to restrict which CAs can issue their certs

Let's Encrypt to revoke 3 million certificates on March 4 due to software bug

Let's Encrypt issued 3,048,289 TLS certificates without checking the CAA field for the requesting domain.

<https://www.zdnet.com/article/lets-encrypt-to-revoke-3-million-certificates-on-march-4-due-to-bug/>

Certificate / Public Key Pinning

Idea: client knows what cert/PK to expect, rejects anything else

How?

- Pre-install some keys
- HPKP (HTTP Public Key Pinning): site can use an HTTP header to let clients know the hash of public key the site will use

What if the site is hacked?

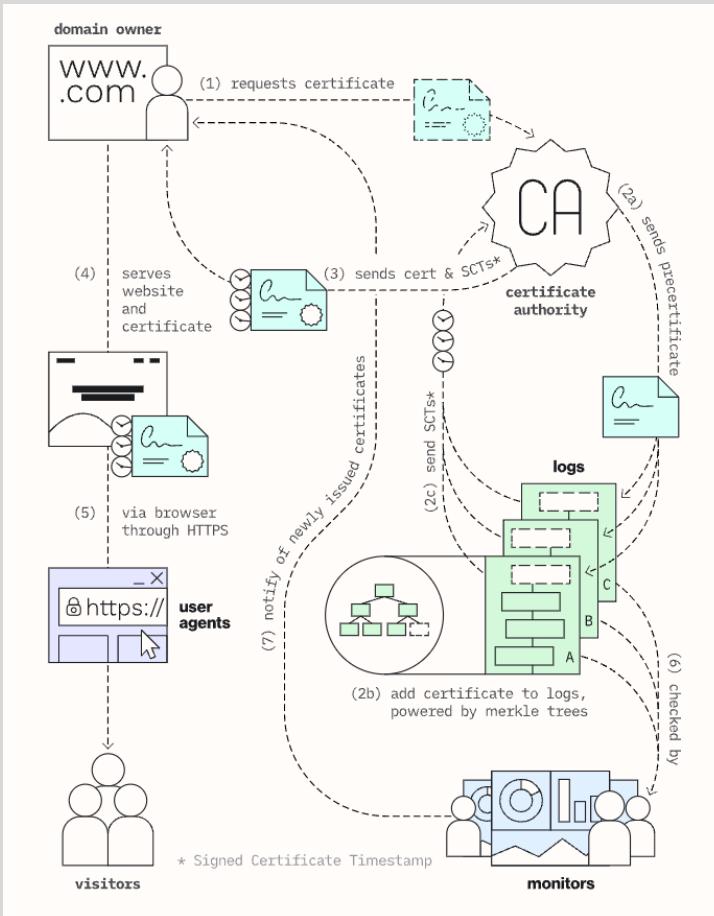
Attacker pins his own key!

Public-Key-Pins:

```
pin-sha256="d6qzRu9zOECb90Uez27xWltNsj0e1Md7GkYYkVoZWmM=";  
pin-sha256="LPJNul+wow4m6DsqxbninhsWHlwfp0JecwQzYpOLmCQ=";  
max-age=259200
```

Note: Chrome never accepted fake DigitNotar certs

Certificate Transparency



Force CAs to log the certificates they sign in a public tamper-evident register

- Server attaches a signed statement from log (SCT) to certificate
- Browser will only use a cert if it is published on two log servers
- Companies can scan logs to look for invalid issuance

Google has been pushing this (+ has its own CA)

- Chrome requires it for EV certs + certs with path to root CA

If certificate is not logged, will users pay attention to browser warnings?

How does this compare to OCSP?

Peeking Through SSL/TLS

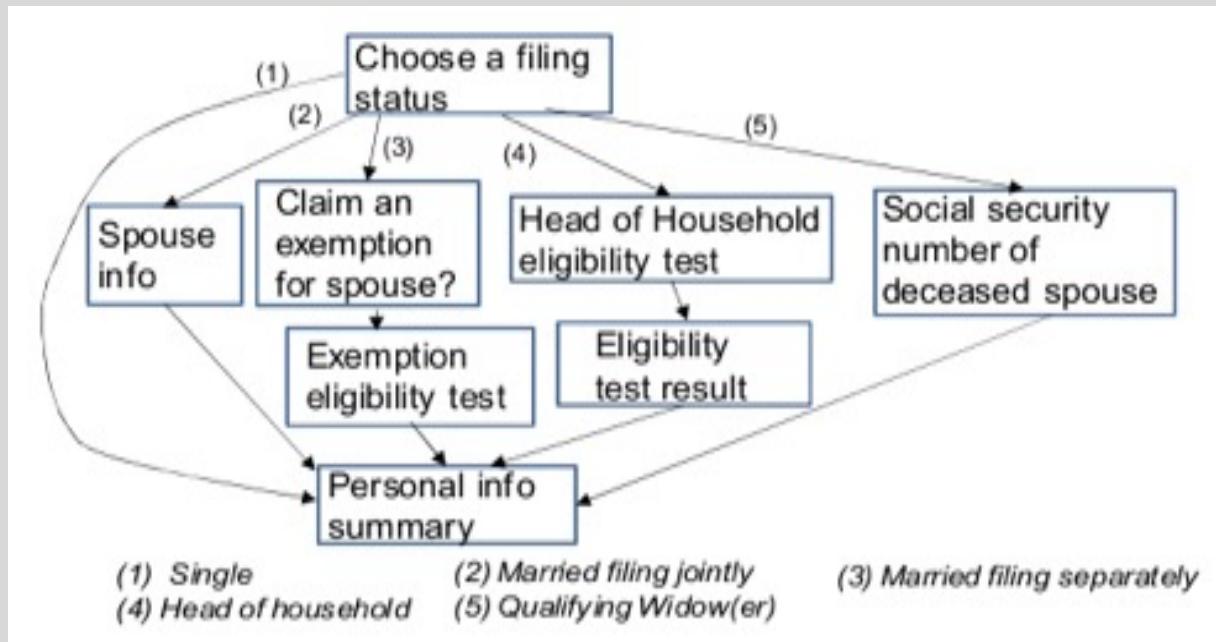
Network traffic reveals length of HTTPS packets

- TLS supports up to 256 bytes of padding

AJAX-rich pages have lots and lots of interactions with the server

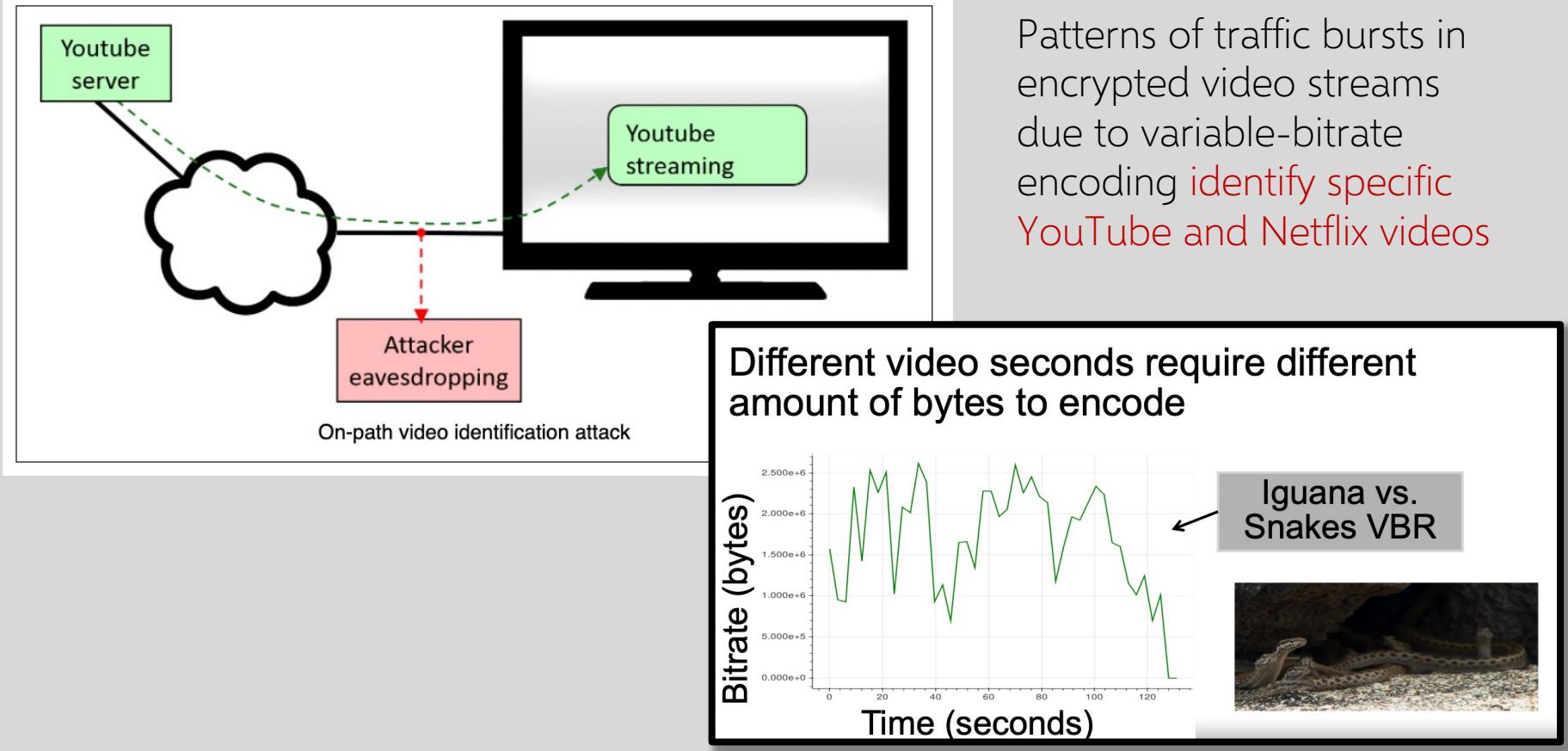
These interactions expose specific internal state of the page

Traffic Analysis: Online Tax Software



Sizes of webpages identify the state of the application, which leaks information about the user's inputs

Traffic Analysis: Video Streaming



Schuster et al. "Beauty and the Burst: Remote Identification of Encrypted Video Streams"