CSE508 Network Security

Stony Brook University

2024-03-26

TLS

Michalis Polychronakis

Stony Brook University

TLS (Transport Layer Security)

- Predecessor: **SSL** (Secure Socket Layer)

Most widely used protocol for encrypted data transmission

- Same basic design, different crypto algorithms

Designed to provide secure communication over the insecure Internet

- Authentication, confidentiality, and integrity

Used in many services and secure versions of protocols

- HTTP, POP, IMAP, SMTP, OpenVPN, CalDAV, CardDAV, LDAP, NNTP, FTP, IRC, SIP, …
- Separate port number: HTTPS: 443, FTPS: 990, IMAPS: 993, DoT: 853, …

History

SSL developed at Netscape

v1: never released

v2 (1994): serious weaknesses

v3 (1995): re-design, basis of what we use today

NETSCAPE®

TLS working group was formed to migrate SSL to IETF

TLS 1.0 (1999): minor differences but incompatible with SSL 3 (different crypto algorithms)

TLS 1.1 (2006): mostly security fixes, TLS extensions

TLS 1.2 (2008): authenticated encryption, more flexible

TLS 1.3 (2018): removal of legacy/weak algorithms, lower latency, perfect forward secrecy, …

Endless cycle of vulnerabilities and improvements

Insecure renegotiation, RC4 weaknesses, compression side channels, padding oracle attacks, buggy implementations, PKI attacks, …

BEAST, CRIME, TIME, Lucky 13, BREACH, POODLE, FREAK, Heartbleed, DROWN, …

Handshake protocol

Negotiate public key crypto algorithms and establish shared secret keys

Authentication (server and optionally client)

Up to TLS 1.2, took 6–10 messages, depending on features used

Record Protocol

Uses the established secret keys to protect the transmitted data

Message transport: `[header|data]` records (16K)

Encryption and integrity: after handshake completion

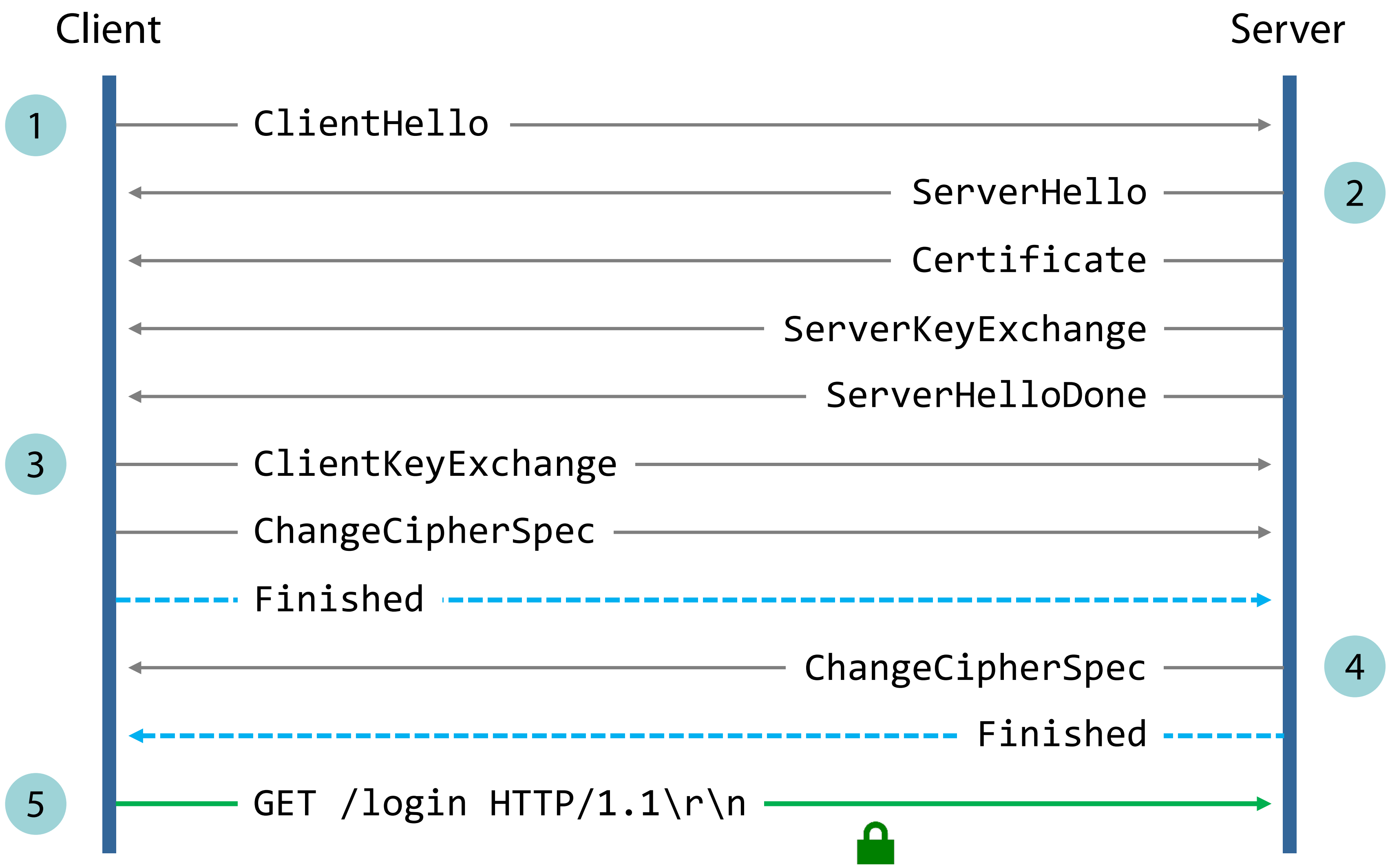
Compression: before encryption… not a good idea

- Side-channel attacks (e.g., CRIME)

Subprotocols: allow for extensibility

- TLS defines four core subprotocols: *handshake, change cipher spec, application data, alert*

TLS 1.2 Handshake (Ephemeral DH)



Cipher Suite Negotiation

ClientHello: *here are the cipher suites I support*

```
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
...
```

ServerHello: *let's use this one*

```
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
```

The server might not support the best of the client's cipher suites

- Offers some other version hoping that the client will accept it

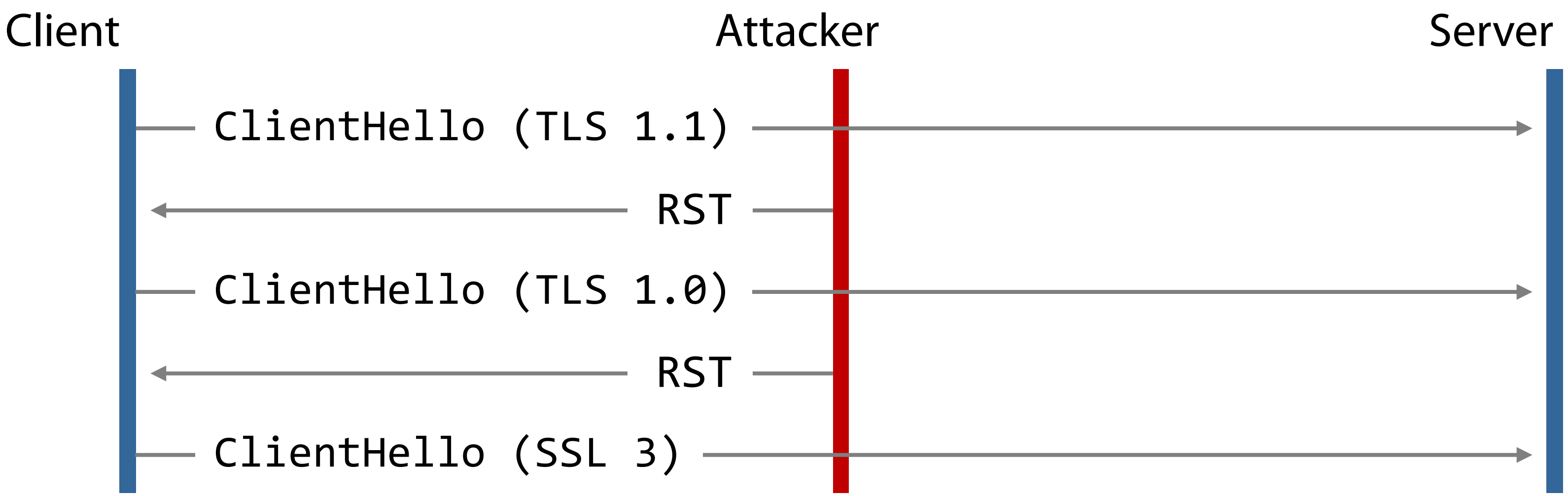
Downgrade Attacks

Force a weaker cipher suite selection through MitM

- SSL 2: no handshake integrity
- SSL 3: protocol rollback protection (still breakable)
- TLS 1.0 and on: additional protections

Due to server bugs and interoperability issues, browsers responded by voluntarily downgrading the protocol upon handshake failure

- Retrying connection with lower SSL/TLS version
- Attackers can exploit this by blocking the initial handshake, or alter the client's list of supported suites



TLS/SSL support history of web browsers

Browser or OS API	Version	Platforms	SSL protocols		TLS protocols				Certificate support			Vulnerability[n 1]						Protocol selection by user[n 2]
			SSL 2.0 (insecure)	SSL 3.0 (insecure)	TLS 1.0 (deprecated)	TLS 1.1 (deprecated)	TLS 1.2	TLS 1.3	EV[n 3][1]	SHA-2[2]	ECDSA[3]	BEAST[n 4]	CRIME[n 5]	POODLE (SSLv3)[n 6]	RC4[n 7]	FREAK[4][5]	Logjam	
Google Chrome (Chrome for Android)[n 8] [n 9]	1–9	Windows (10+) macOS (10.15+) Linux Android (8.0+) iOS (14+) ChromeOS	Disabled by default	Yes	Yes	No	No	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected[10]	Vulnerable (HTTPS)	Vulnerable	Vulnerable	Vulnerable (except Windows)	Vulnerable	Yes[n 10]
	10–20		No[11]	Yes	Yes	No	No	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected	Vulnerable (HTTPS/ SPDY)	Vulnerable	Vulnerable	Vulnerable (except Windows)	Vulnerable	Yes[n 10]
	21		No	Yes	Yes	No	No	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected	Mitigated[12]	Vulnerable	Vulnerable	Vulnerable (except Windows)	Vulnerable	Yes[n 10]
	22–29		No	Yes	Yes	Yes[13]	No[13][14][15][16]	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected	Mitigated	Vulnerable	Vulnerable	Vulnerable (except Windows)	Vulnerable	Temporary[n 11]
	30–32		No	Yes	Yes	Yes	Yes[14][15][16]	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected	Mitigated	Vulnerable	Vulnerable	Vulnerable (except Windows)	Vulnerable	Temporary[n 11]
	33–37		No	Yes	Yes	Yes	Yes	No	Yes (only desktop)	Requires SHA-2 compatible OS[2]	Needs ECC compatible OS[3]	Not affected	Mitigated	Partly mitigated[n 12]	Lowest priority[19][20] [21]	Vulnerable (except Windows)	Vulnerable	Temporary[n 11]
	38, 39		No	Yes	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Partly mitigated	Lowest priority	Vulnerable (except Windows)	Vulnerable	Temporary[n 11]
	40		No	Disabled by default[18][22]	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Mitigated[n 13]	Lowest priority	Vulnerable (except Windows)	Vulnerable	Yes[n 14]
	41, 42		No	Disabled by default	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Mitigated	Lowest priority	Mitigated	Vulnerable	Yes[n 14]
	43		No	Disabled by default	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Mitigated	Only as fallback[n 15][23]	Mitigated	Vulnerable	Yes[n 14]
	44–47		No	No[24]	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Not affected	Only as fallback[n 15]	Mitigated	Mitigated[25]	Temporary[n 11]
	48, 49		No	No	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Needs ECC compatible OS[3]	Not affected	Mitigated	Not affected	Disabled by default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]
	50–53		No	No	Yes	Yes	Yes	No	Yes (only desktop)	Yes	Yes	Not affected	Mitigated	Not affected	Disabled by default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]
	54–66		No	No	Yes	Yes	Yes	Disabled by default (draft version)	Yes (only desktop)	Yes	Yes	Not aff						
	67–69		No	No	Yes	Yes	Yes	Yes (draft version)	Yes (only desktop)	Yes	Yes	Not affected	Mitigated	Not affected	default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]
	70–83		No	No	Yes	Yes	Yes	Yes	Yes (only desktop)	Yes	Yes	Not affected	Mitigated	Not affected	Disabled by default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]
	84–90		No	No	Warn by default	Warn by default	Yes	Yes	Yes (only desktop)	Yes	Yes	Not affected	Mitigated	Not affected	Disabled by default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]
	91–121 / ESC 122, 123		No	No	No[28]	No[28]	Yes	Yes	Yes (only desktop)	Yes	Yes	Not affected	Mitigated	Not affected	Disabled by default[n 16][26] [27]	Mitigated	Mitigated	Temporary[n 11]

SSL 3.0, TLS 1.0, and TLS 1.1 are now completely removed by most browsers

https://en.wikipedia.org/wiki/Version_history_for_TLS/SSL_support_in_web_browsers

TLS 1.2 Session Resumption

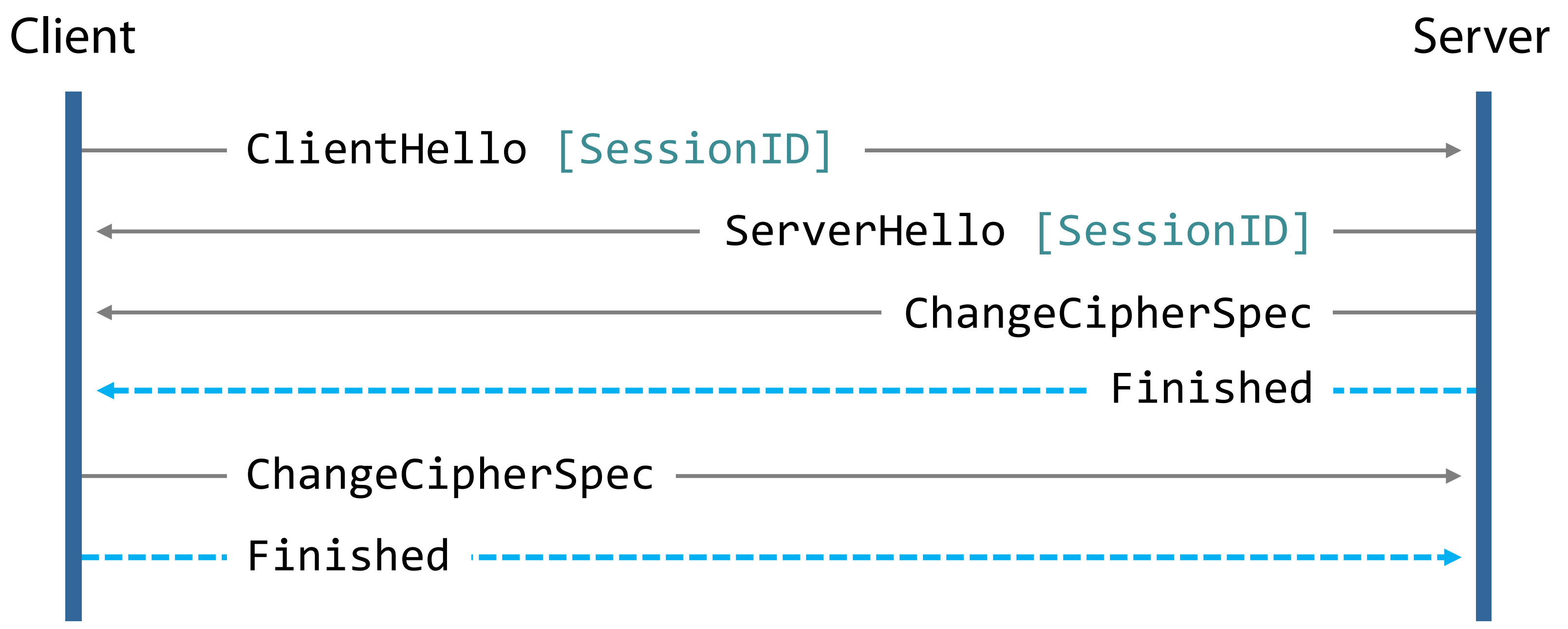
Full handshake: 6-10 messages and two network round-trips

- Along with CPU-intensive crypto operations, cert validation, …

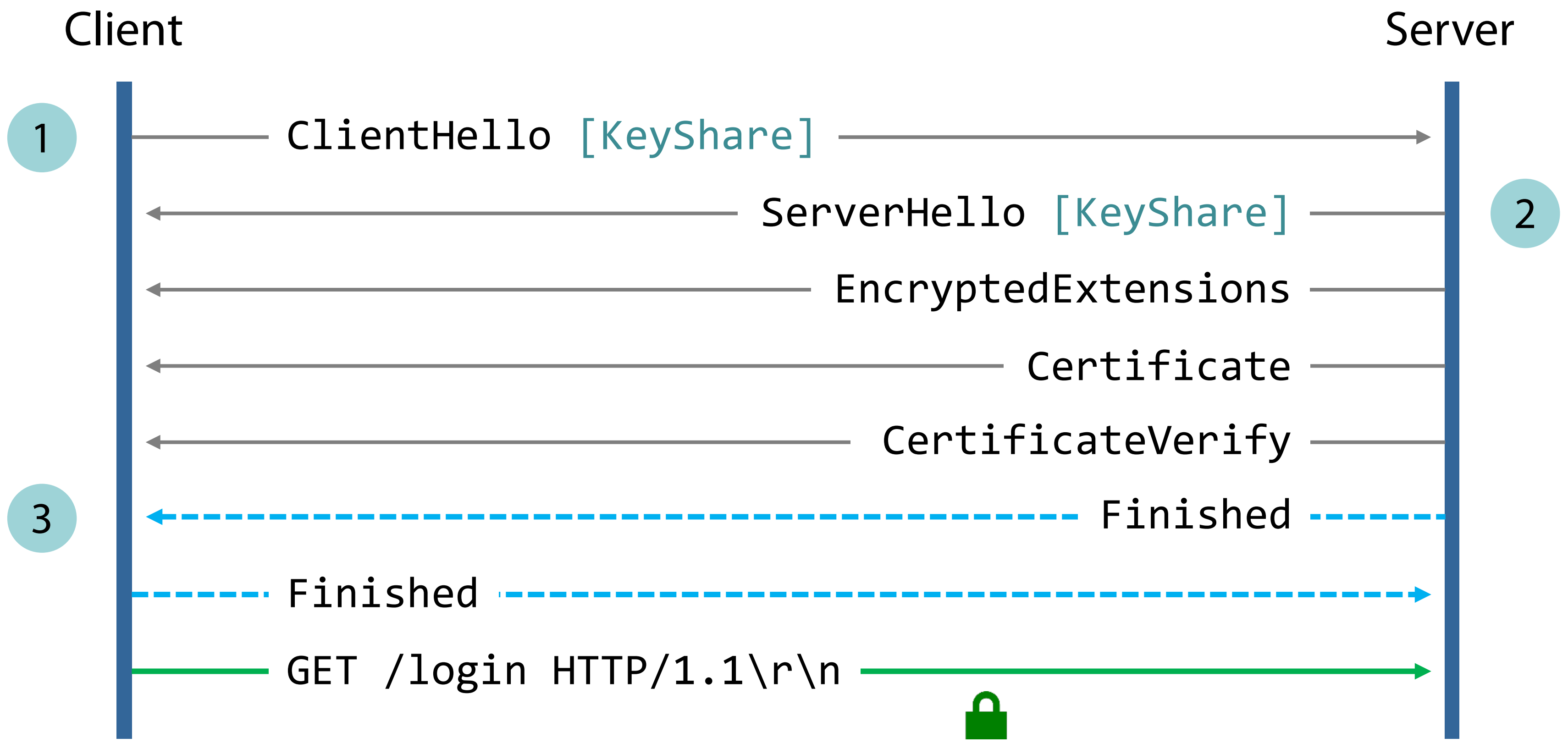
Avoid re-negotiation by remembering security parameters

- Server assigns and sends a unique *Session ID* as part of `ServerHello`
- In future connections, the client sends the Session ID to resume the session

Alternative: *session tickets* (all state is kept at client)



TLS 1.3 Handshake (Ephemeral DH)



Latest draft supports even zero-RTT handshakes

Clients include encrypted data in the initial messages based on config. ID previously sent by server

TLS Server (and Client) Authentication

After handshake completion, the client knows it can "trust" the information in the server's certificate

- Assuming it trusts the issuing certificate authority

TLS certs are based on the X.509 PKI standard

How is the certificate associated with the server?

- *Common Name (CN):* ***server's hostname***

Certificate-based authentication is also supported for clients

- Highly-secure web services, some VPN services, cloud applications, …

Rarely used in practice for user authentication

- Common alternative: username + password over TLS connection

Certificate Fields

Version: v1 (basic), v2 (additional fields), v3 (extensions)

Serial Number: high-entropy integer

Signature Algorithm: encryption and hash algorithm used to sign the cert

Issuer: contains the *distinguished name (DN)* of the certificate issuer

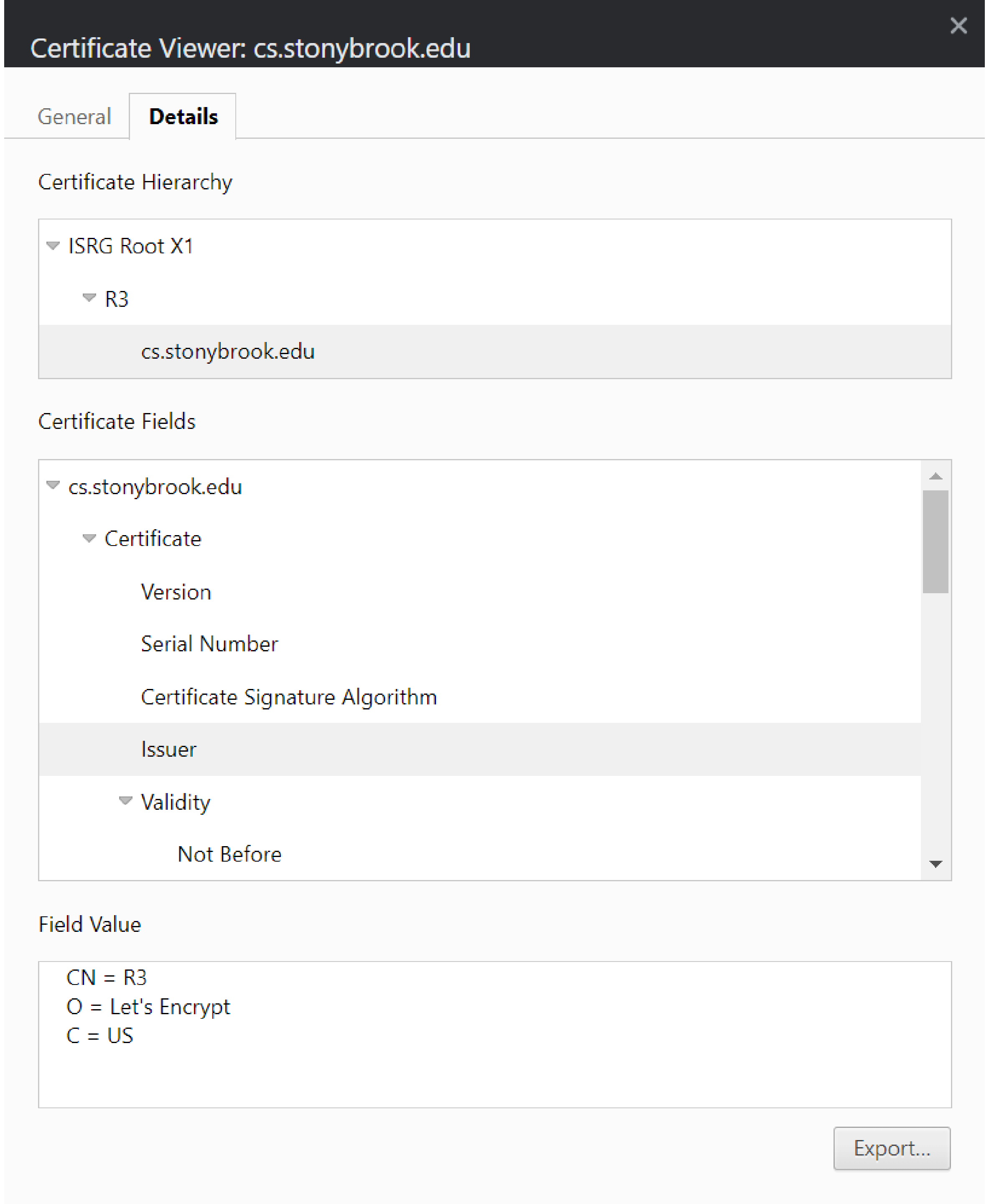
Validity: starting and ending date of validity period

Subject: DN of the entity associated with the certificate's public key

- Deprecated in favor of the Subject Alternative Name (SAN) extension: DNS name, IP address, or URI (also supports binding to multiple identities)

Public Key: The subject's public key

Signature


Certificate Viewer: cs.stonybrook.edu
General
Details
Certificate Hierarchy
ISRG Root X1
R3
cs.stonybrook.edu
Certificate Fields
cs.stonybrook.edu
Certificate
Version
Serial Number
Certificate Signature Algorithm
Issuer
Validity
Not Before
Field Value
CN = R3
O = Let's Encrypt
C = US
Export...

Certificate Chains

Trust anchors: operating systems and browsers are pre-configured with trusted root certificates

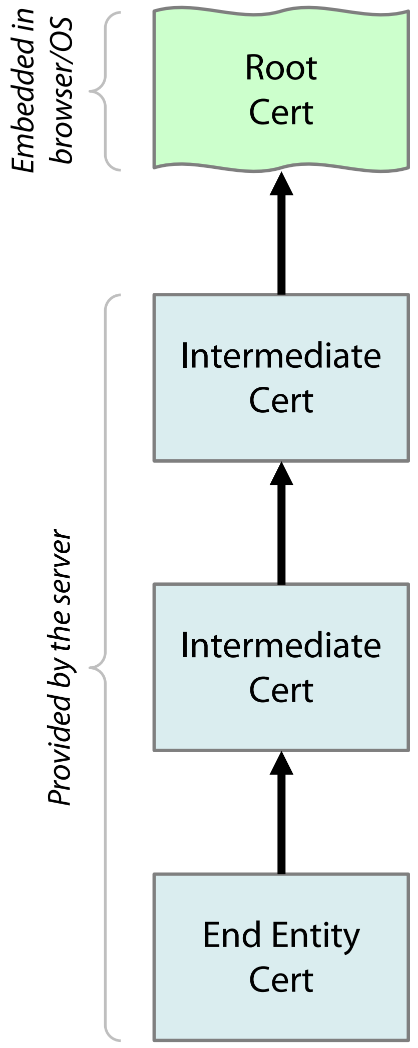
- System/public store: used by OS, browsers, …
- More can be added in the local/private cert store: vendor-specific certs, MitM certs for content inspection filters/AVs, …

Server provides a *chain* of certificates

- A certificate from an intermediate CA is trusted if there is a valid chain of trust all the way back to a trusted root CA

Any CA can issue and sign certificates for any subject

- The system is only as secure as the weakest certificate authority…
- Certificate Authority Authorization (CAA): can be used to restrict which CAs can issue certificates for a particular domain



Qualys. SSL Labs

SSL Report: www.cs.stonybrook.edu (23.185.0.4)

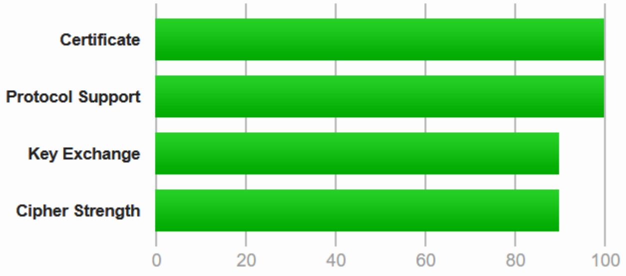
Assessed on: Mon, 11 Mar 2024 18:40:24 UTC | Hide | Clear cache

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.

This server supports TLS 1.3.

Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1

Subject	cs.stonybrook.edu Fingerprint SHA256: 4cd2591703fe3242248d300e30ad8a18b1b030550d3af84c70ede1053c530f76 Pin SHA256: tb0gpmFSHp26fT3bLpiBcyAYMke2WdQY0e4cnllWi4E=

SSL Server Test: www.cs.stonybr

https://www.ssllabs.com/ssltest/analyze.html?d=www.cs.stonybrook.edu&s=23.185.0.4

Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1

Subject	cs.stonybrook.edu Fingerprint SHA256: 4cd2591703fe3242248d300e30ad8a18b1b030550d3af84c70ede1053c530f76 Pin SHA256: tb0gpmFSHp26fT3bLpiBcyAYMke2WdQY0e4cnllWi4E=
Common names	cs.stonybrook.edu
Alternative names	cs.stonybrook.edu www.cs.stonybrook.edu www.cs.sunysb.edu
Serial Number	036318e8fa07c3c517af871800fd0cc3b6ea
Valid from	Fri, 02 Feb 2024 20:02:35 UTC
Valid until	Thu, 02 May 2024 20:02:34 UTC (expires in 1 month and 21 days)
Key	RSA 2048 bits (e 65537)
Weak key (Debian)	No
Issuer	R3 AIA: http://r3.i.lencr.org/
Signature algorithm	SHA256withRSA
Extended Validation	No
Certificate Transparency	**Yes (certificate)**
OCSP Must Staple	No
Revocation information	OCSP OCSP: http://r3.o.lencr.org
Revocation status	Good (not revoked)
DNS CAA	No (more info)
Trusted	**Yes** **Mozilla Apple Android Java Windows**

Additional Certificates (if supplied)

Certificates provided	3 (3994 bytes)
Chain issues	None

#2

	R3

Certification Authority Authorization (CAA)

By default, certificate authorities (CA) are allowed to issue certificates for any domain name (after they validate control of that domain name)

- Bugs in CA's domain validation process
- Compromised CAs

CAA: Specify which CAs are authorized to issue certificates for a domain

- Reduce the "attack surface" of CAs that could otherwise issue unauthorized certificates

Implemented as a special DNS resource record: CAA

- Specifies the allowed CAs, policy flags, …

CAs are required to check CAA records and comply with their directives

- Third parties monitoring CA behavior might check newly issued certificates against the domain's CAA records and identify violators

Certificate Revocation

Allow revocation of compromised or no longer needed certificates

Certificate revocation list (CRL)

- Signed list of all revoked certificates that have not yet expired
- Main problem: lists tend to be large, making real-time lookups slow
- Can the attacker block connectivity to the CA's server?
- CRLSets (Chrome): revocation list pushed to the browser as a *software update*

Online Certificate Status Protocol (OCSP)

- Obtain the revocation status of a *single* certificate ➔ faster
- But the latency, security, and privacy issues still remain
- OCSP stapling (Firefox): server embeds OCSP response directly into the TLS handshake (soft-fail issue remains: an adversary can suppress the OCSP response)

HTTPS

Most common use of TLS: *almost all web traffic is now encrypted*

All major roadblocks of the past are not a problem anymore

- Crypto is expensive, HTTPS needs more CPU cycles → native hardware support
- Mixed HTTP+HTTPS content → almost all third-party ad networks/widgets/etc. now support HTTPS
- Virtual hosting was initially incompatible → solved with TLS 1.1 through the *Server Name Indication (SNI)* extension
- Needs expertise and certs cost $$$$ → certs are now free and easy to deploy and maintain through letsencrypt.org

Let's Encrypt

Firesheep In Wolves' Clothing: ✕

https://techcrunch.com/2010/10/24/firesheep-in-wolves-clothing-app-lets-you-hack-into-twitter-facebook-accounts-easi

TC

Join Extra Crunch

Login

Search

TC Early Stage 2021

Startups

Videos

Audio

Newsletters

Extra Crunch

The TC List

Advertise

Events

—

More

Firesheep In Wolves' Clothing: Extension Lets You Hack Into Twitter, Facebook Accounts Easily

Contributor 11:24 PM EST • October 24, 2010

Comment

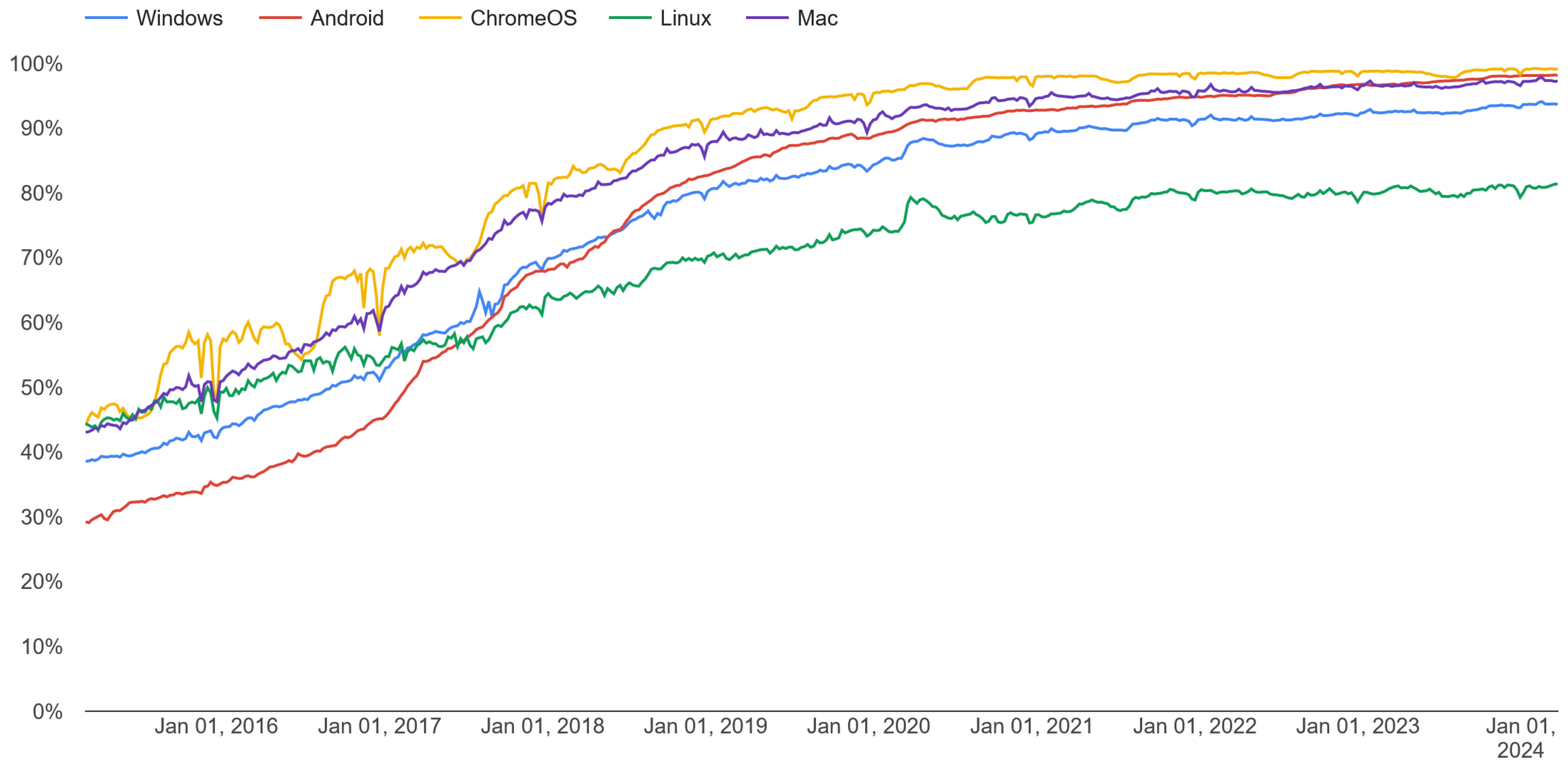
It seems like every time Facebook amends its privacy policy, the web is up in arms. The truth is, Facebook's well publicized privacy fight is nothing compared to the vulnerability of all unsecured HTTP sites — that includes Facebook, Twitter and many of the web's most popular destinations.

Developer Eric Butler has exposed the soft underbelly of the web with his new Firefox extension, Firesheep, which will let you essentially eavesdrop on any open Wi-Fi network and capture users' cookies.

As Butler explains in his post, "As soon as anyone on the network visits an insecure website known to Firesheep, their name and photo will be displayed" in the window. All you have to do is double click on

Facebook HTTP → HTTPS
2010: HTTPS only for login
2010: Firesheep released
2011: Optional full HTTPS
2013: HTTPS on by default

Percentage of pages loaded over HTTPS in Chrome by platform



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

Browser Security Indicators (state until circa 2015)

Convey information about the security of a page

Locks, shields, keys, green bars…

“This page was fetched using SSL”

Page content was not viewed or altered by a network adversary

Certificate is valid (e.g. not expired), issued by a CA trusted by the browser, and the subject name matches the URL’s domain

“This page uses an invalid certificate”

“Parts of the page are not encrypted”

“The legal entity operating this web site is known”

Extended Validation (EV) certificates

Browser Security Indicators (state until circa 2015)

Convey informa[illegible] about the security of [illegible]ge

- Locks, shields, k[illegible]n bars…

“This page was fetch[illegible]g SSL”

- Page content was not vi[illegible]lt[illegible] network adversary
- Certificate is valid (e.g. not e[illegible]ed by a CA trusted by the browser, and the subject name matches the U[illegible]

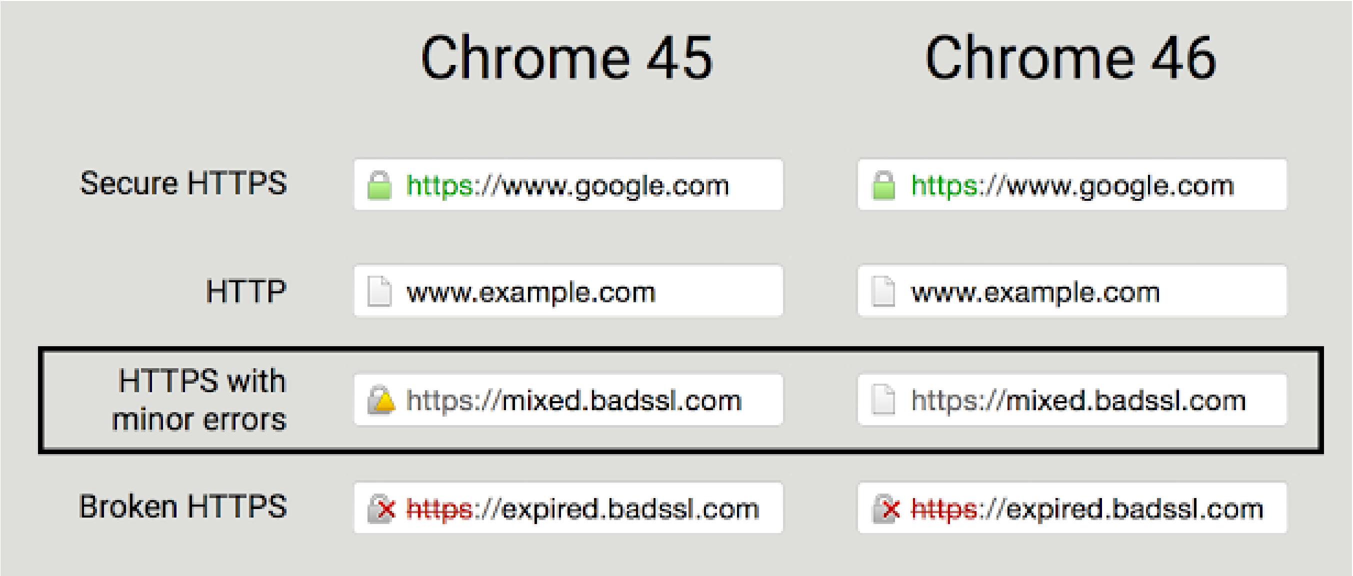
“This page uses an inva[illegible]ica[illegible]

“Parts of the page a[illegible]encrypted”

“The legal entit[illegible]ting this web site is [illegible]”

- Extended Validation (EV) certificates

Mixed Content Warning is Unnecessary (2015)



Basically the same in terms of security

Non-HTTPS traffic is a vulnerability!

MitM/MotS attacks on the HTTP part are trivial

https://googleonlinesecurity.blogspot.com/2015/10/simplifying-page-security-icon-in-chrome.html

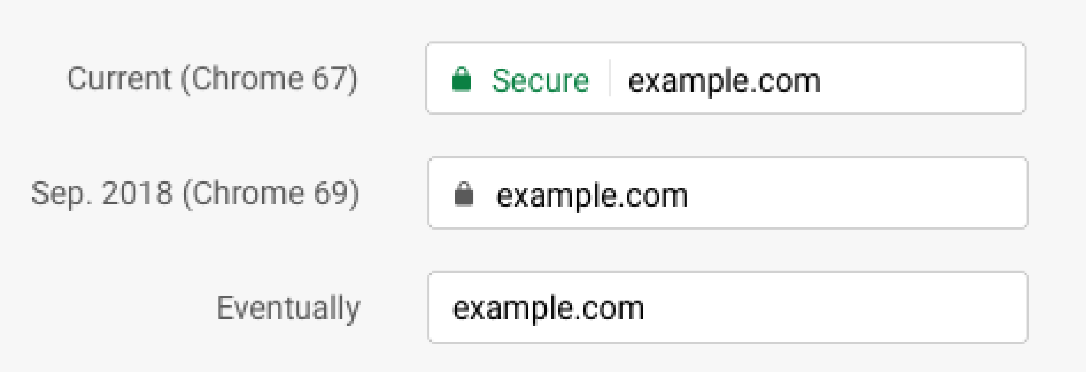
Mark HTTP as Not Secure (2018)

Majority of traffic is HTTPS → negative indicator for HTTP

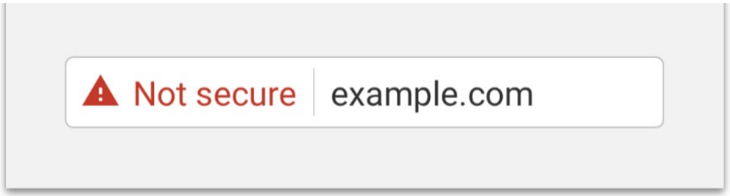
Before: mark HTTPS as secure

Majority of traffic was HTTP → positive indicator for HTTPS

HTTPS

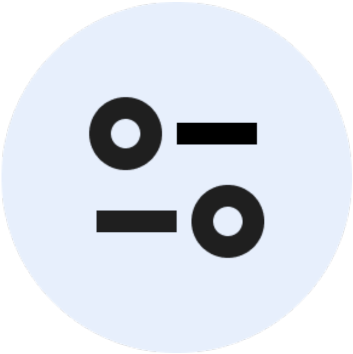


HTTP



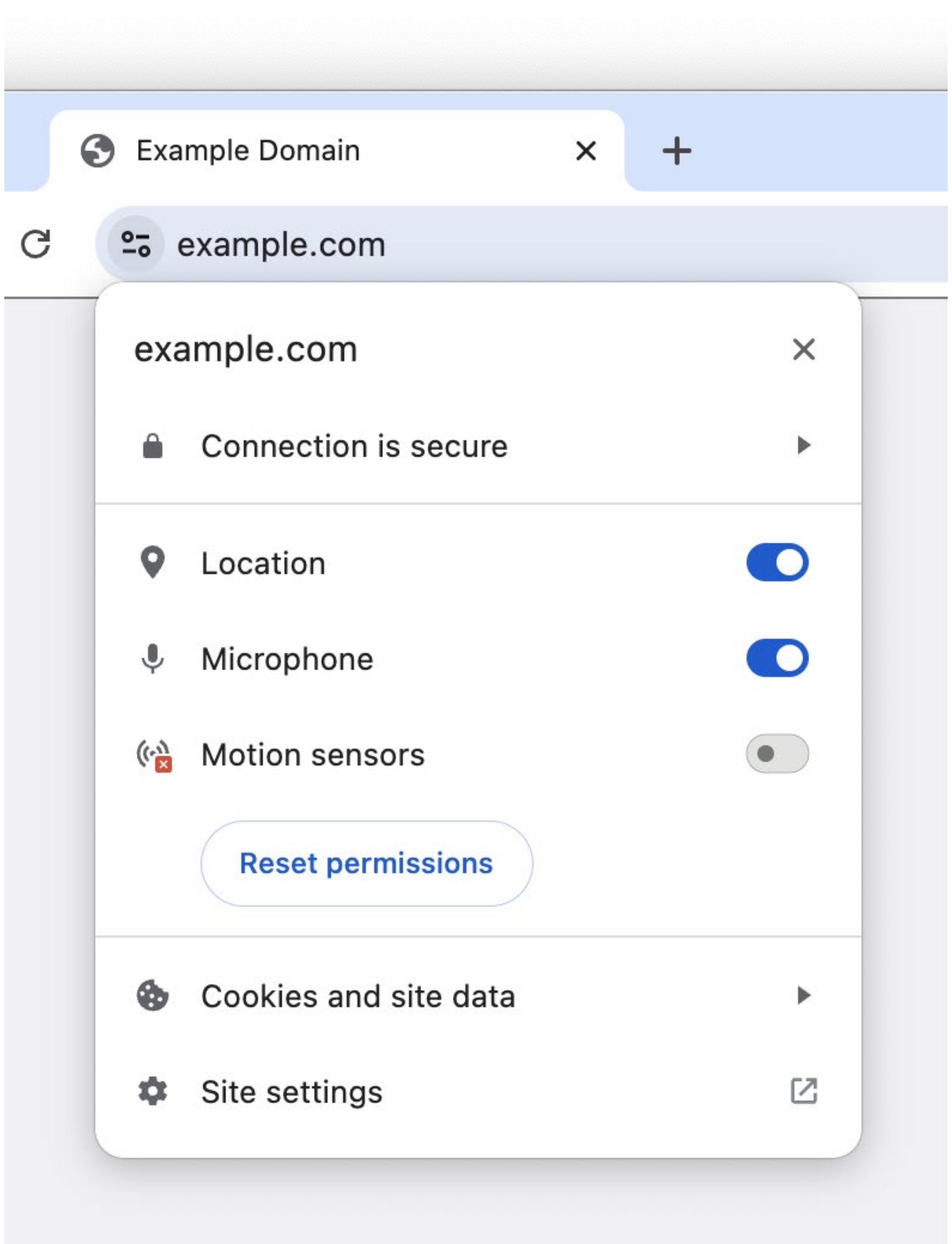
https://blog.chromium.org/2018/05/evolving-chromes-security-indicators.html

Lock Icon Is Gone (2023)



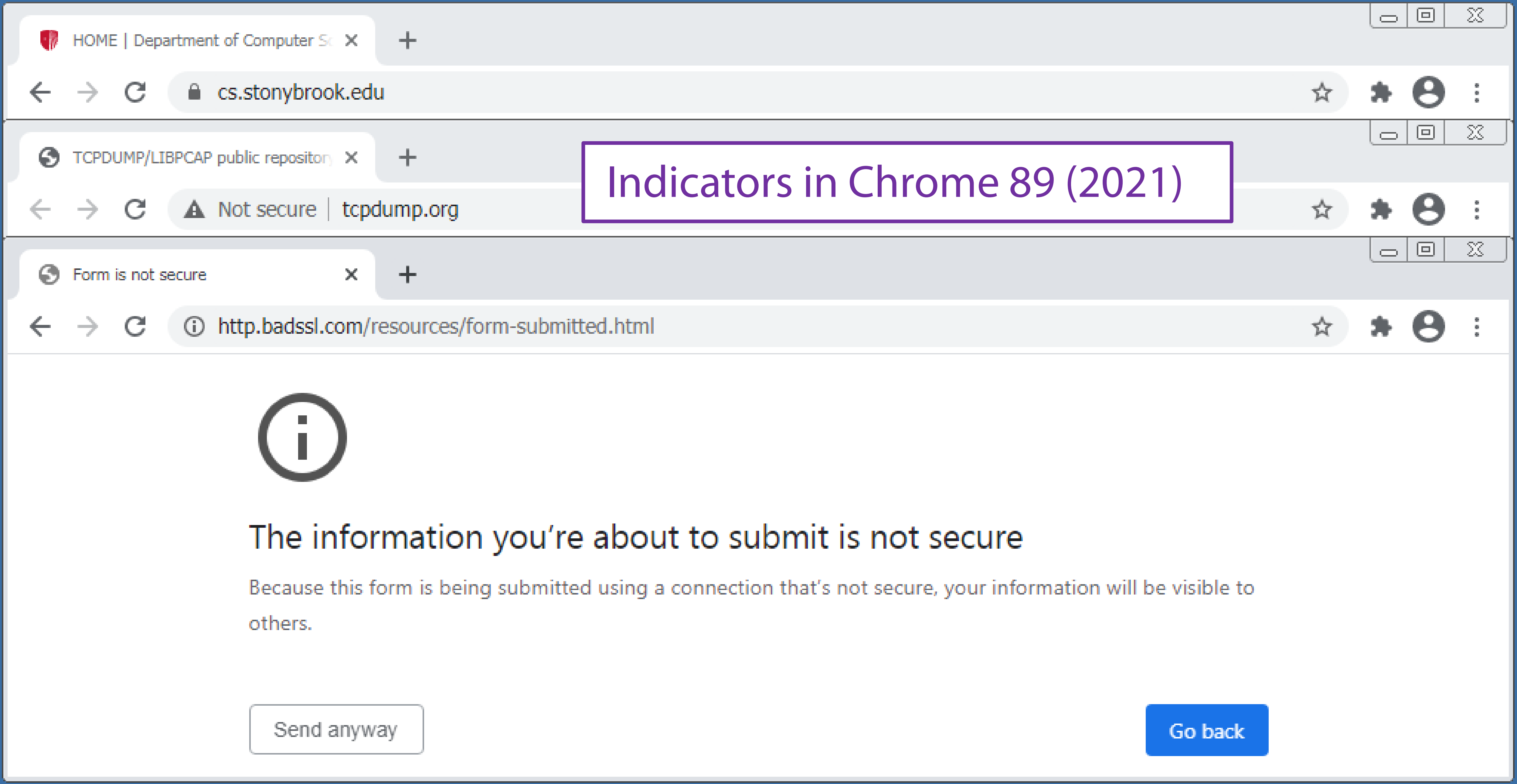
“Tune icon”

- Does not imply "trustworthy"
- Is more obviously clickable
- Is commonly associated with settings or other controls

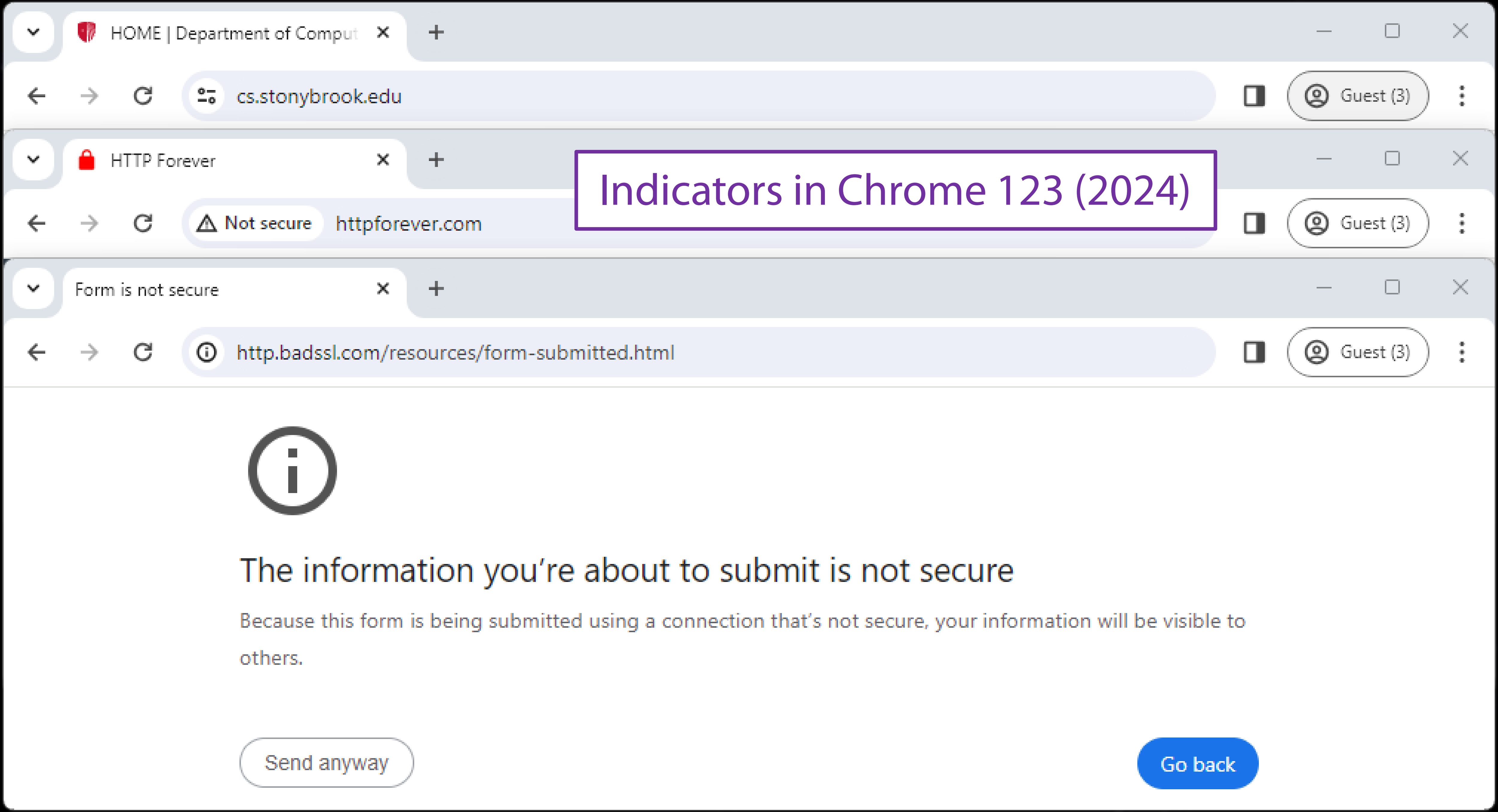


https://blog.chromium.org/2023/05/an-update-on-lock-icon.html

Indicators in Chrome 89 (2021)



Indicators in Chrome 123 (2024)



SSL Stripping

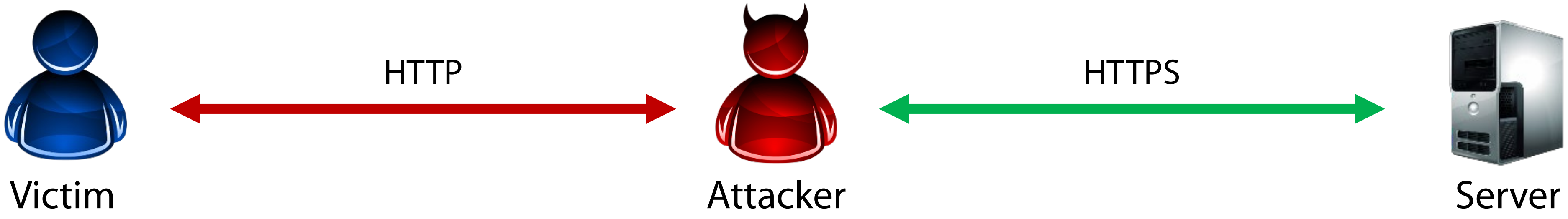
Browsing sessions often start with a plain HTTP request

- Web sites used to switch to HTTPS only for login or checkout
- Example: Facebook in 2010 (optional full HTTPS in 2011, HTTPS by default in 2013)
- Users type addresses without specifying `https://`
- Browser connects over HTTP *by default* ➔ site may redirect to HTTPS

SSLstrip [Moxie Marlinspike, Black Hat DC 2009]

- MitM attack to prevent redirection to HTTPS
- Watch for **HTTPS** redirects and links, and map them to **HTTP** links
- …or homograph-similar *valid* HTTPS links (similar to DNS poisoning):
- `https://www.bank.com.attacker.com`

SSL stripping




```
Location: http://…
 <a href="http://…">
<form action="http://…">
```

```
Location: https://…
 <a href="https://…">
<form action="https://…">
```

~~Missing lock icon~~ "Not secure" warning or different domain, but who is going to notice?

HSTS (HTTP Strict Transport Security)

Defense against SSL stripping and other similar issues

- Force the use of HTTPS instead of HTTP ***before*** accessing a resource
- Treat all errors (e.g., invalid certificate, mixed content, plain HTTP) as fatal: do not allow users to access the web page

Servers implement HSTS policies by supplying an extra HTTP header

- **`Strict-Transport-Security: max-age=31536000`**
- *"From now on and for the next year, use only HTTPS for requests to this domain"*

An instance of trust on first use (TOFU)

- Problem: the initial request *remains unprotected* because it is sent over HTTP
- Solution: **HSTS preloading:** browsers come preloaded with a list of known HSTS sites

transport_security_state_static.js ×

source.chromium.org/chromium/chromium/src/+/master:net/http/transport_security_state_static.json

Chromium Code Search

Search for code or files

chromium/src > master > net/http/transport_security_state_static.json

Edit code

Files Outline

http
transport_security_state_source.h
transport_security_state_static.json
transport_security_state_static.pins
transport_security_state_static.templa
transport_security_state_static_fuzzer
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_static_unittes
transport_security_state_test_util.cc
transport_security_state_test_util.h
transport_security_state_unittest.cc
url_security_manager.cc
url_security_manager.h
url_security_manager_posix.cc
url_security_manager_unittest.cc
url_security_manager_win.cc
webfonts_histogram.cc
webfonts_histogram.h
webfonts_histogram_unittest.cc

transport_security_state_static.json

Find Links View in Related files Blame

```
    // START OF LEGACY MANUAL CUSTOM ENTRIES
    { "name": "www.paypal.com", "policy": "custom", "mode": "force-https" },
    { "name": "paypal.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.elanex.biz", "policy": "custom", "mode": "force-https" },
    { "name": "www.noisebridge.net", "policy": "custom", "mode": "force-https" },
    { "name": "neg9.org", "policy": "custom", "mode": "force-https" },
    { "name": "factor.cc", "policy": "custom", "mode": "force-https" },
    { "name": "aladdinschools.appspot.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.paycheckrecords.com", "policy": "custom", "mode": "force-https" },
    { "name": "lastpass.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.lastpass.com", "policy": "custom", "mode": "force-https" },
    { "name": "entropia.de", "policy": "custom", "mode": "force-https" },
    { "name": "www.entropia.de", "policy": "custom", "mode": "force-https" },
    { "name": "logentries.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.logentries.com", "policy": "custom", "mode": "force-https" },
    { "name": "dropcam.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.dropcam.com", "policy": "custom", "mode": "force-https" },
    { "name": "epoxate.com", "policy": "custom", "mode": "force-https" },
    { "name": "torproject.org", "policy": "custom", "mode": "force-https", "pins": "tor" },
    { "name": "blog.torproject.org", "policy": "custom", "mode": "force-https", "include_subdomains": true, "pins": "tor" },
    { "name": "check.torproject.org", "policy": "custom", "mode": "force-https", "include_subdomains": true, "pins": "tor" },
    { "name": "www.torproject.org", "policy": "custom", "mode": "force-https", "include_subdomains": true, "pins": "tor" },
    { "name": "dist.torproject.org", "policy": "custom", "mode": "force-https", "include_subdomains": true, "pins": "tor" },
    { "name": "ledgerscope.net", "policy": "custom", "mode": "force-https" },
    { "name": "www.ledgerscope.net", "policy": "custom", "mode": "force-https" },
    { "name": "greplin.com", "policy": "custom", "mode": "force-https" },
    { "name": "www.greplin.com", "policy": "custom", "mode": "force-https" },
```

History References

Firefox 83 introduces HTTPS-Only Mode

Christoph Kerschbaumer, Julian Gaibler, Arthur Edelstein and Thyla van der Merwe | November 17, 2020

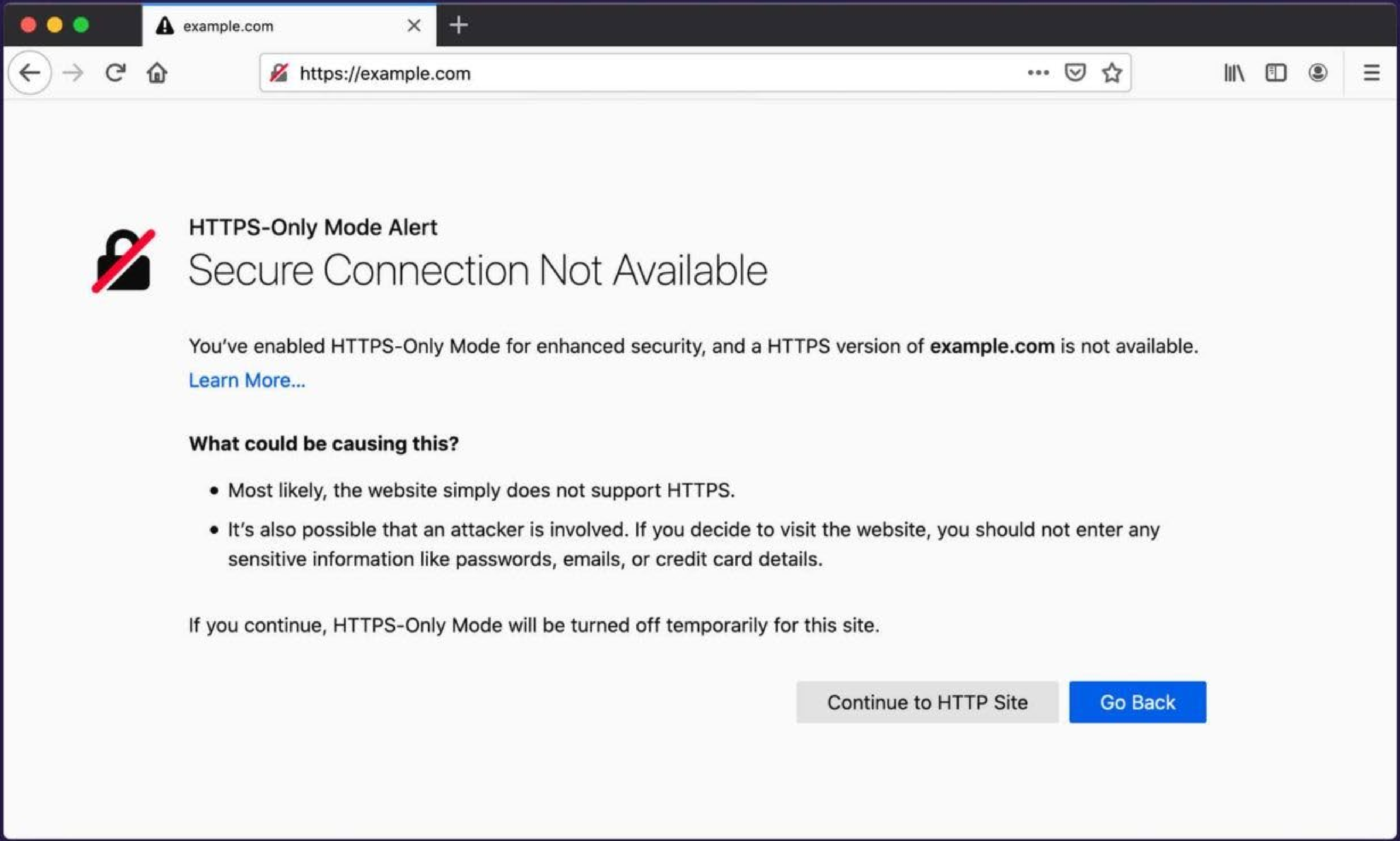
Security on the web matters. Whenever you connect to a web page and enter a password, a credit card number, or other sensitive information, you want to be sure that this information is kept secure. Whether you are writing a personal email or reading a page on a medical condition, you don't want that information leaked to eavesdroppers on the network who have no business prying into your personal communications.

That's why Mozilla is pleased to introduce HTTPS-Only Mode, a brand-new security feature available in Firefox 83. When you enable HTTPS-Only Mode:

- Firefox attempts to establish fully secure connections to every website, and
- Firefox asks for your permission before connecting to a website that doesn't support secure connections.

How HTTPS-Only Mode works

The Hypertext Transfer Protocol (HTTP) is a fundamental protocol through which web browsers and websites communicate. However, data transferred by the regular HTTP protocol is unprotected and transferred in cleartext, such that attackers are able to view, steal, or even tamper with the transmitted


example.com
https://example.com
HTTPS-Only Mode Alert
Secure Connection Not Available
You've enabled HTTPS-Only Mode for enhanced security, and a HTTPS version of example.com is not available.
Learn More...
What could be causing this?
Most likely, the website simply does not support HTTPS.
It's also possible that an attacker is involved. If you decide to visit the website, you should not enter any sensitive information like passwords, emails, or credit card details.
If you continue, HTTPS-Only Mode will be turned off temporarily for this site.
Continue to HTTP Site
Go Back

Chromium Blog: A safer default

https://blog.chromium.org/2021/03/a-safer-default-for-navigation-https.html

Chromium Blog

News and developments from the open source browser project

Chrome 90 (2021)

A safer default for navigation: HTTPS

Tuesday, March 23, 2021

Starting in version 90, Chrome's address bar will use *https://* by default, improving privacy and even loading speed for users visiting websites that support HTTPS. Chrome users who navigate to websites by manually typing a URL often don't include "http://" or "https://". For example, users often type "example.com" instead of "https://example.com" in the address bar. In this case, if it was a user's first visit to a website, Chrome would previously choose *http://* as the default protocol[1]. This was a practical default in the past, when much of the web did not support HTTPS.

Chrome will now default to HTTPS for most typed navigations that don't specify a protocol[2]. HTTPS is the more secure and most widely used scheme in Chrome on all major platforms. In addition to being a clear security and privacy improvement, this

Search blog ...

Labels

Archive

Feed

Follow @ChromiumDev

Give us feedback in our Product Forums.

MitM is Still Possible…

Rogue certificates

- Most governments have a trusted root CA planted in our systems
- Attackers may break into CAs and forge certificates

Pre-planted/generated certificates

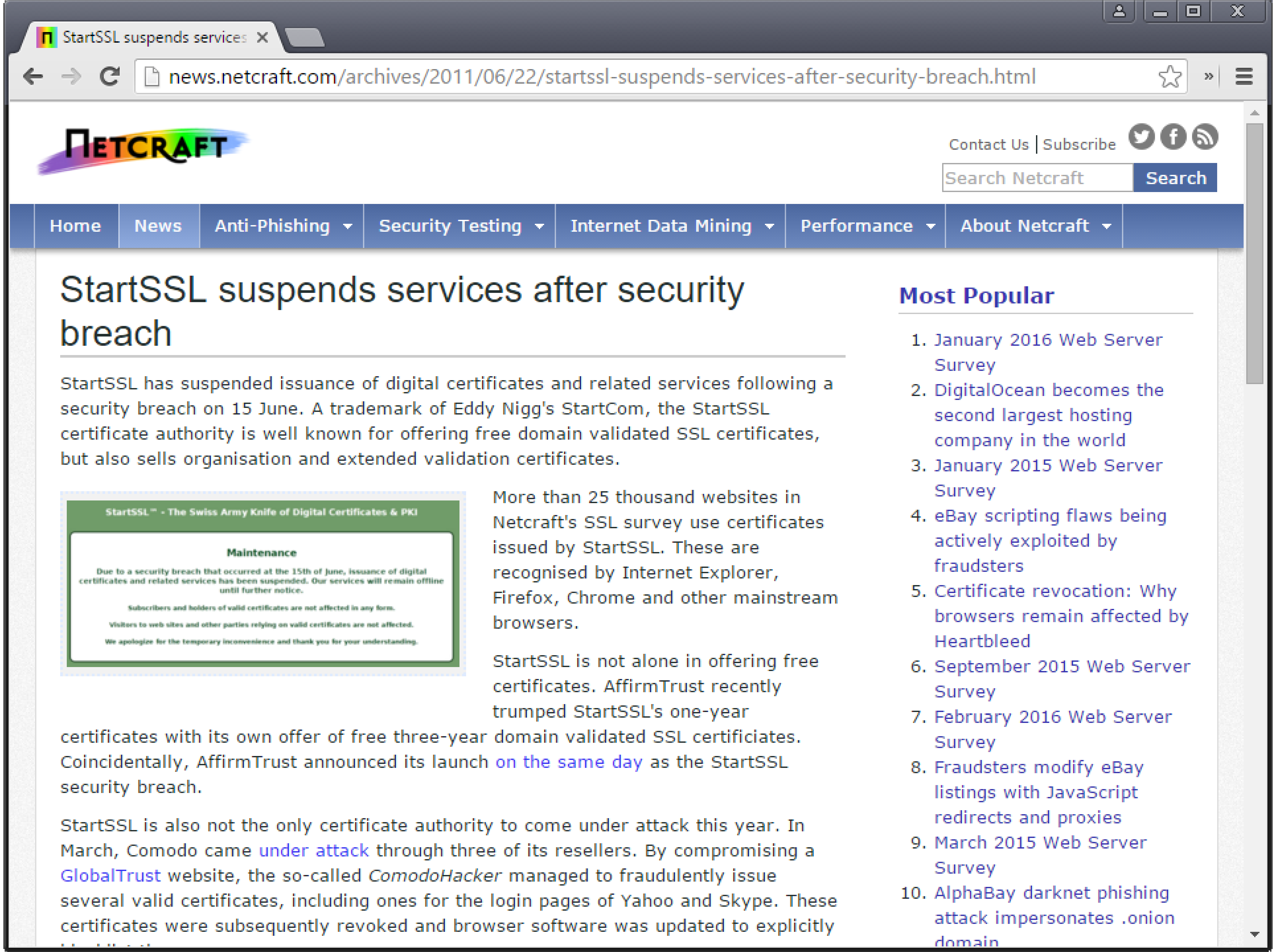
- Default static keys: Lenovo, Dell, anti-malware software, …
- Low entropy during key generation: repeated or factorable keys

Self-signed certificates

- If desperate… will trigger scary browser warning

Exploitation of certificate validation flaws

- Programming errors while checking date, hostname, …



StartSSL suspends services

news.netcraft.com/archives/2011/06/22/startssl-suspends-services-after-security-breach.html

NETCRAFT

Contact Us | Subscribe

Search Netcraft | Search

Home | News | Anti-Phishing | Security Testing | Internet Data Mining | Performance | About Netcraft

StartSSL suspends services after security breach

StartSSL has suspended issuance of digital certificates and related services following a security breach on 15 June. A trademark of Eddy Nigg's StartCom, the StartSSL certificate authority is well known for offering free domain validated SSL certificates, but also sells organisation and extended validation certificates.

StartSSL™ - The Swiss Army Knife of Digital Certificates & PKI

Maintenance

Due to a security breach that occurred at the 15th of June, issuance of digital certificates and related services has been suspended. Our services will remain offline until further notice.

Subscribers and holders of valid certificates are not affected in any form.

Visitors to web sites and other parties relying on valid certificates are not affected.

We apologize for the temporary inconvenience and thank you for your understanding.

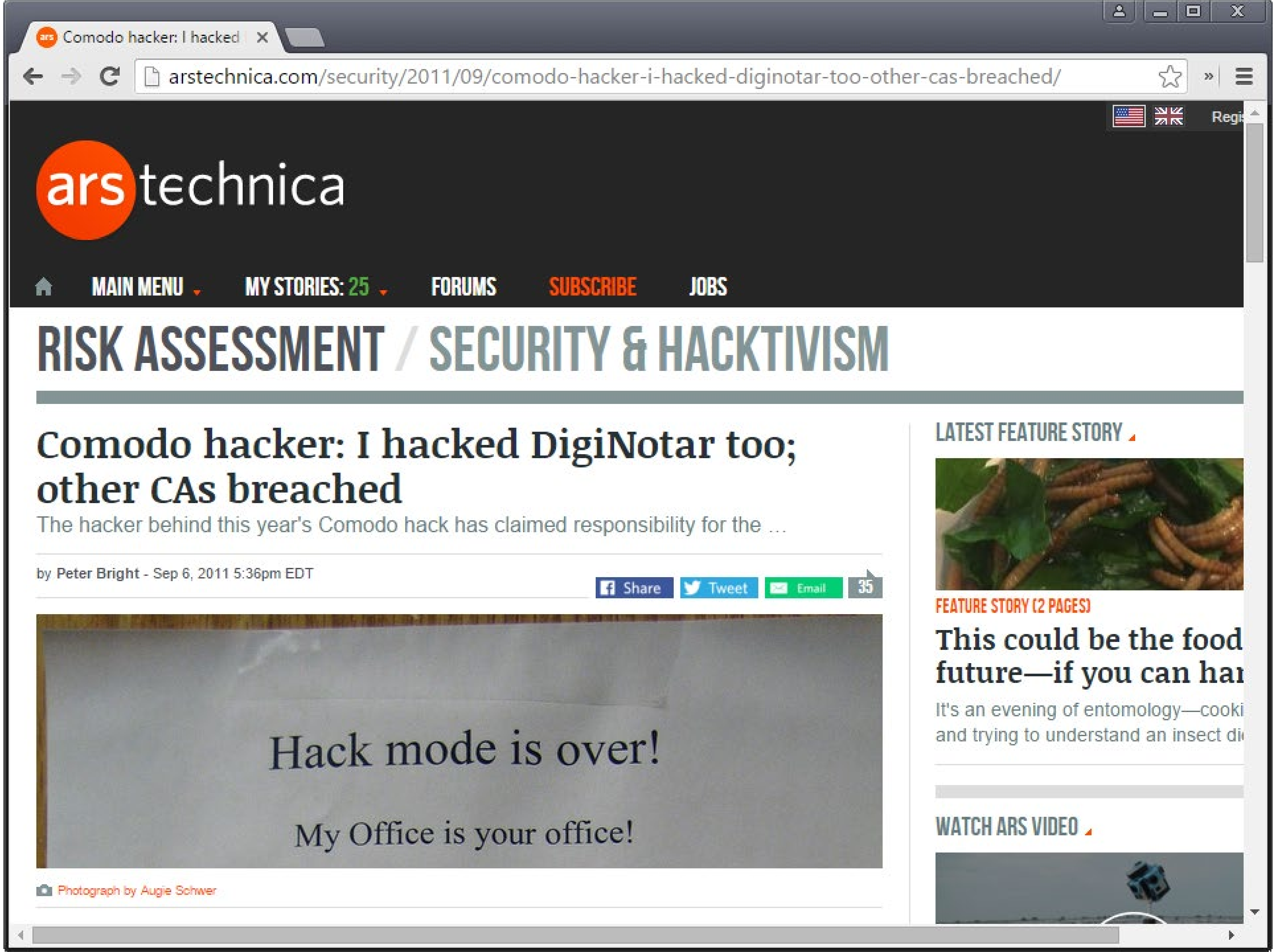
More than 25 thousand websites in Netcraft's SSL survey use certificates issued by StartSSL. These are recognised by Internet Explorer, Firefox, Chrome and other mainstream browsers.

StartSSL is not alone in offering free certificates. AffirmTrust recently trumped StartSSL's one-year certificates with its own offer of free three-year domain validated SSL certificiates. Coincidentally, AffirmTrust announced its launch on the same day as the StartSSL security breach.

StartSSL is also not the only certificate authority to come under attack this year. In March, Comodo came under attack through three of its resellers. By compromising a GlobalTrust website, the so-called *ComodoHacker* managed to fraudulently issue several valid certificates, including ones for the login pages of Yahoo and Skype. These certificates were subsequently revoked and browser software was updated to explicitly

Most Popular

1. January 2016 Web Server Survey
2. DigitalOcean becomes the second largest hosting company in the world
3. January 2015 Web Server Survey
4. eBay scripting flaws being actively exploited by fraudsters
5. Certificate revocation: Why browsers remain affected by Heartbleed
6. September 2015 Web Server Survey
7. February 2016 Web Server Survey
8. Fraudsters modify eBay listings with JavaScript redirects and proxies
9. March 2015 Web Server Survey
10. AlphaBay darknet phishing attack impersonates .onion domain


Comodo hacker: I hacked
arstechnica.com/security/2011/09/comodo-hacker-i-hacked-diginotar-too-other-cas-breached/
ars technica
MAIN MENU
MY STORIES: 25
FORUMS
SUBSCRIBE
JOBS
RISK ASSESSMENT / SECURITY & HACKTIVISM
Comodo hacker: I hacked DigiNotar too; other CAs breached
The hacker behind this year's Comodo hack has claimed responsibility for the …
by Peter Bright - Sep 6, 2011 5:36pm EDT
Share
Tweet
Email
35
Hack mode is over!
My Office is your office!
Photograph by Augie Schwer
LATEST FEATURE STORY
FEATURE STORY (2 PAGES)
This could be the food
future—if you can ha
It's an evening of entomology—cooki
and trying to understand an insect di
WATCH ARS VIDEO

Trustwave to escape 'deatl

www.theregister.co.uk/2012/02/14/trustwave_analysis/

The Register®

Biting the hand that feeds IT

DATA CENTER SOFTWARE NETWORKS SECURITY INFRASTRUCTURE DEVOPS BUSINESS HARDWARE SCIENCE BOOTNOTES FORUMS

Security

Trustwave to escape 'death penalty' for SSL skeleton key

Moz likely to spare certificate-confession biz same fate as DigiNotar

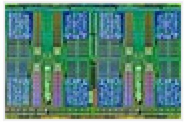
14 Feb 2012 at 09:28, John Leyden 12 10

Analysis Trustwave's admission that it issued a digital "skeleton key" that allowed an unnamed private biz to spy on SSL-encrypted connections within its corporate network has sparked a fiery debate about trust on the internet.

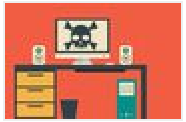
Trustwave, an SSL certificate authority, confessed to supplying a subordinate root certificate as part of an information security product that allowed a customer to monitor employees' web communications - even if the staffers relied on HTTPS. Trustwave said the man-in-the-middle (MitM) gear was designed both to be tamper-proof and to work only within its unnamed client's compound. Despite these precautions, Trustwave now admits that the whole approach was misconceived and would not be repeated. In addition, it revoked the offending certificate.

Trustwave came clean without the need for pressure beforehand. Even so its action have split security experts and prompted calls on Mozilla's Bugzilla security list to remove the Trustwave root certificate

Most read

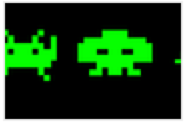


AMD to fix slippery hypervisor-busting its CPU microcode

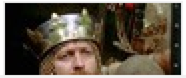


First working Apple ransomware infects Transmission BitTo app downloads

Amazon douses fla vows to restore Fire fondleslab encrypti

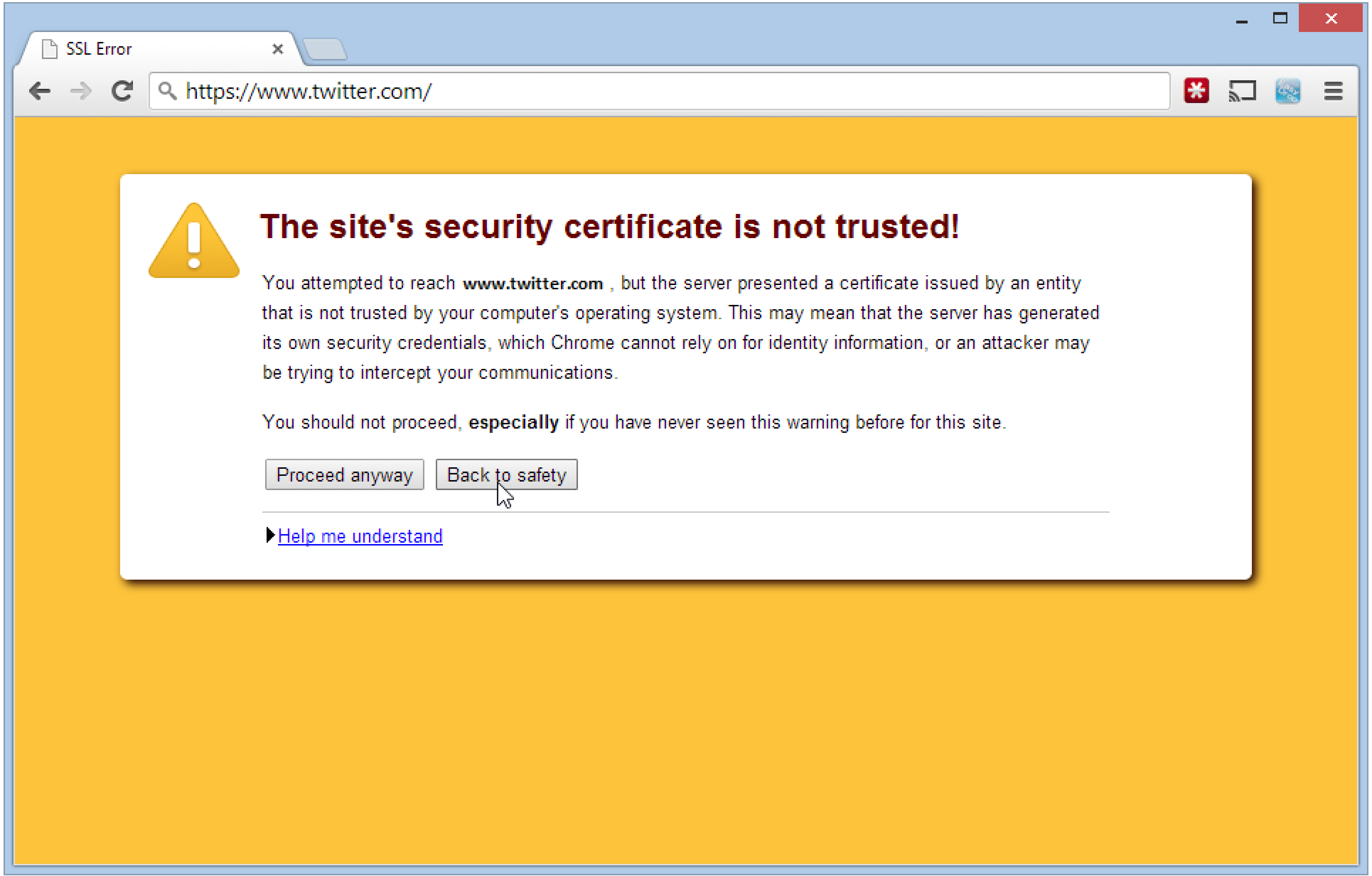


MAME goes fully F

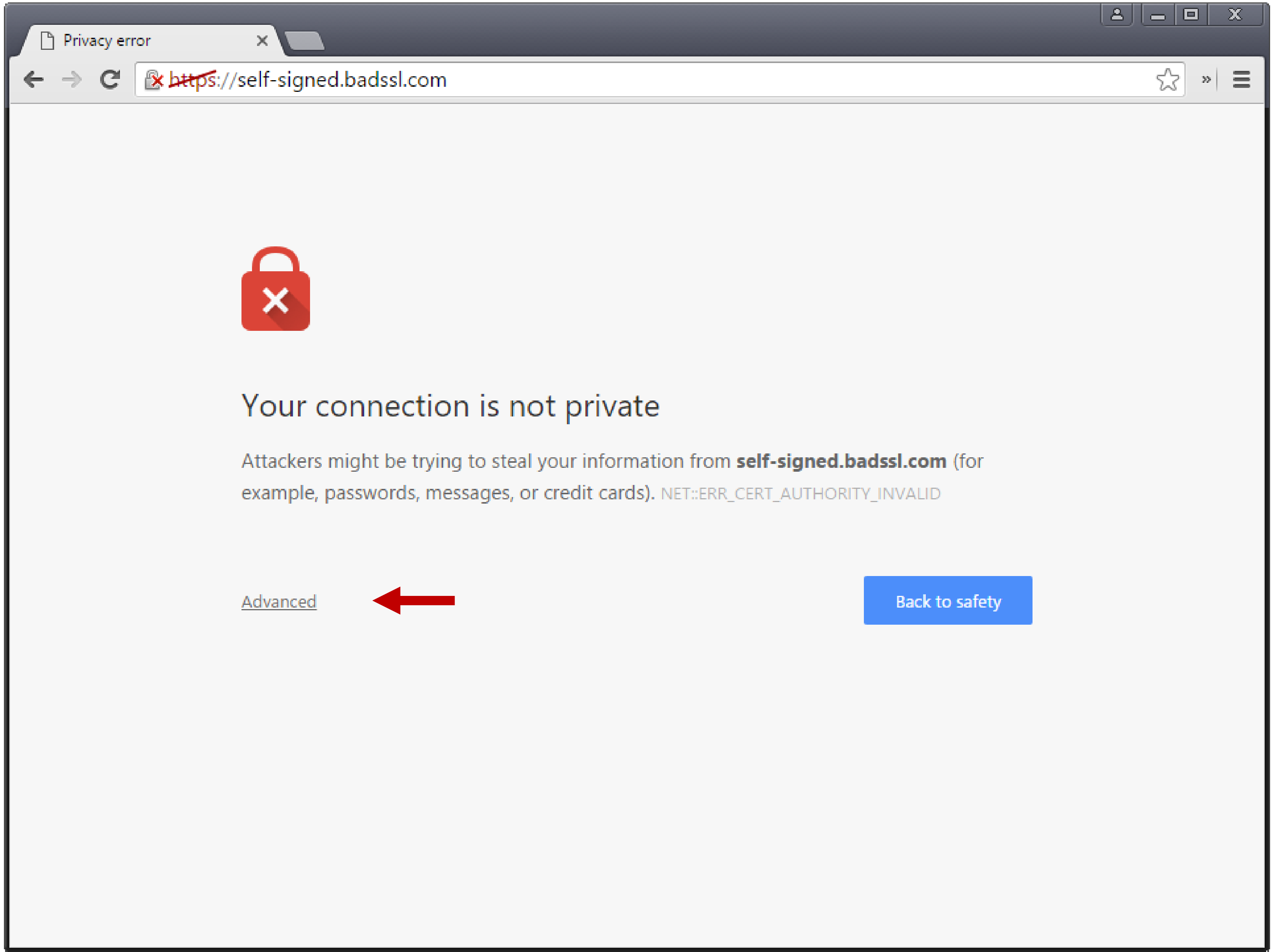


McAfee gaffe a qui kill for enterprising

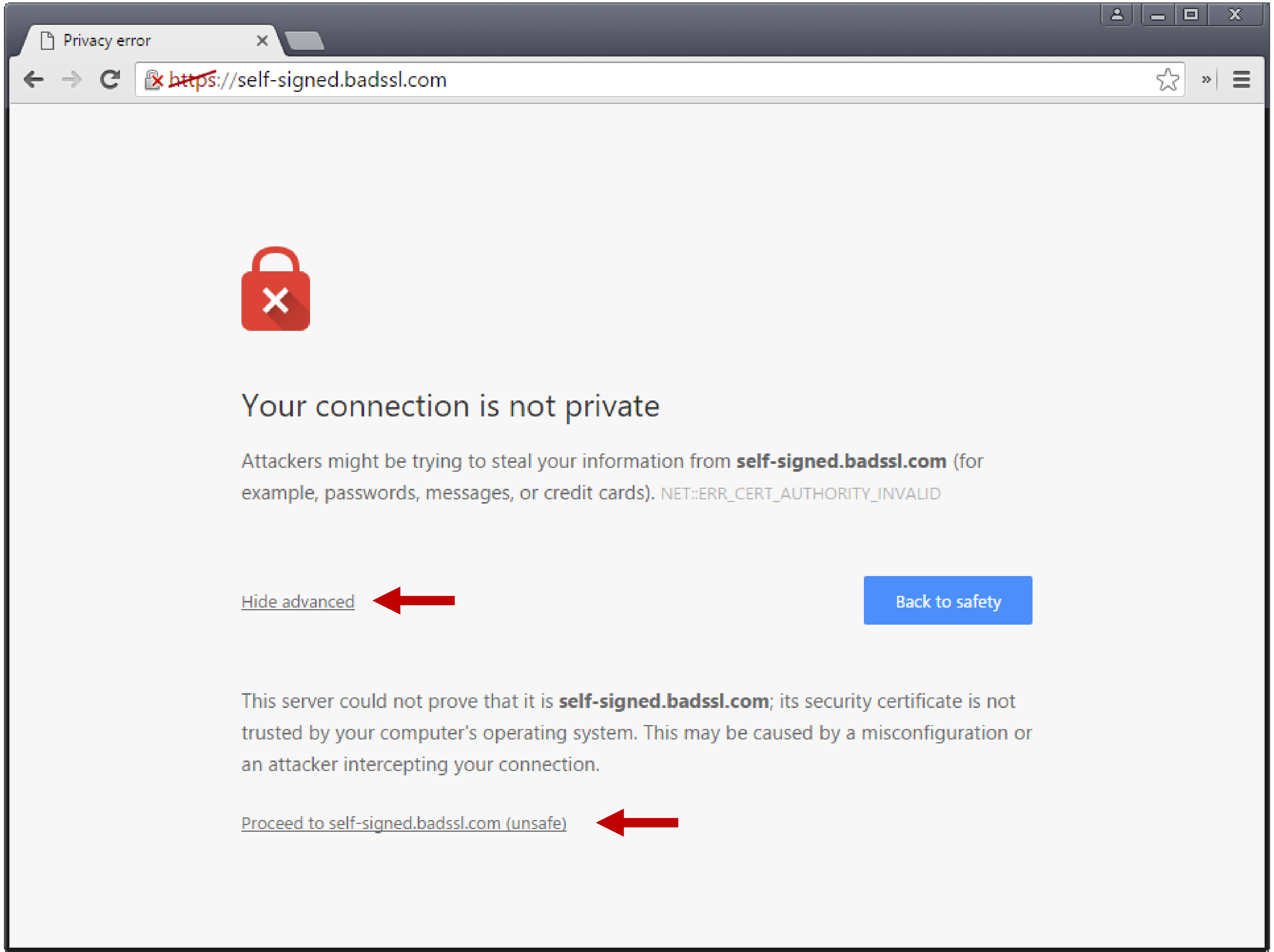
Self-signed Certificate Warning in the Past: One click away…



Self-signed Certificate Warning Now: Two clicks away…



Self-signed Certificate Warning Now: Two clicks away…



GOTO FAIL

iOS 7.0.6 signature verification error

Legitimate-looking TLS certificates with mismatched private keys were unconditionally accepted…

```
    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;
    ...

fail:
    SSLFreeBuffer(&signedHashes);
    SSLFreeBuffer(&hashCtx);
    return err;
```

?!!?!?!?

Check never executed

HPKP (HTTP Public Key Pinning)

Prevent *certificate forgery*: strong form of web site authentication

Browser knows the *valid* public keys of a particular website

- If a seemingly valid chain does not include at least one known pinned key, the cert is rejected
- Doesn't apply for *private* root certificates (would break preconfigured proxies, anti-malware, content filters, …)

Many incidents involving rogue certificates were discovered after browsers started rolling out pinning

Similar deployment as HSTS

- TOFU: HTTP response header
- Built-in pins in browsers

Must be used very carefully – things can go wrong

- HPKP suicide: site can be bricked if keys are lost/stolen
- RansomPKP: compromise the server and push a malicious HPKP key

HPKP (HTTP Public Key Pinning)

Prevent *certificate forgery*: strong form of web site authentication

Browser knows [illegible] public keys of a pa[illegible] website

- If a seemingly valid [illegible]es not include at le[illegible]own pinned key, the cert is rejected
- Doesn't apply for *priva*[illegible]rtificates (w[illegible]k preconfigured proxies, anti-malware, content filters, …)

Many incidents involving r[illegible]tes were discovered after browsers started rolling out pinning

Similar deployment as HST[illegible]

- TOFU: HTTP response he[illegible]
- Built-in pins in browse[illegible]

Must be used ver[illegible]ully – things ca[illegible]rong

- HPKP suicide: s[illegible]e bricked if keys are lost/sto[illegible]
- RansomPKP: comp[illegible]omise the server and push a malici[illegible]s HPKP key

Deprecated in favor of ***Certificate Transparency*** *and the* ***Expect-CT*** *header*

Google Online Security Blc

https://googleonlinesecurity.blogspot.com/2013/01/enhancing-digital-certificate-security.html

Google Security Blog

The latest news and insights from Google on security and safety on the Internet

Enhancing digital certificate security

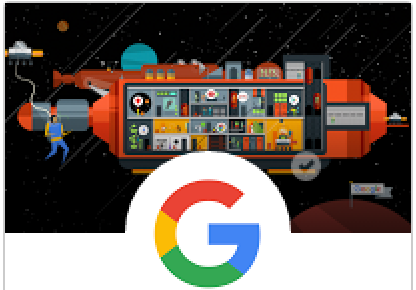
January 3, 2013

Posted by Adam Langley, Software Engineer

Late on December 24, Chrome detected and blocked an unauthorized digital certificate for the "*.google.com" domain. We investigated immediately and found the certificate was issued by an intermediate certificate authority (CA) linking back to TURKTRUST, a Turkish certificate authority. Intermediate CA certificates carry the full authority of the CA, so anyone who has one can use it to create a certificate for any website they wish to impersonate.

In response, we updated Chrome's certificate revocation metadata on

Search blog ...



Google

google.com/+google

News and updates on Google's products, technology and more

G+ Follow +1

+ 11,007,947

Certificate Transparency

Public monitoring and auditing of certificates

- Identify mistakenly or maliciously issued certificates and rogue CAs

Certificate logs

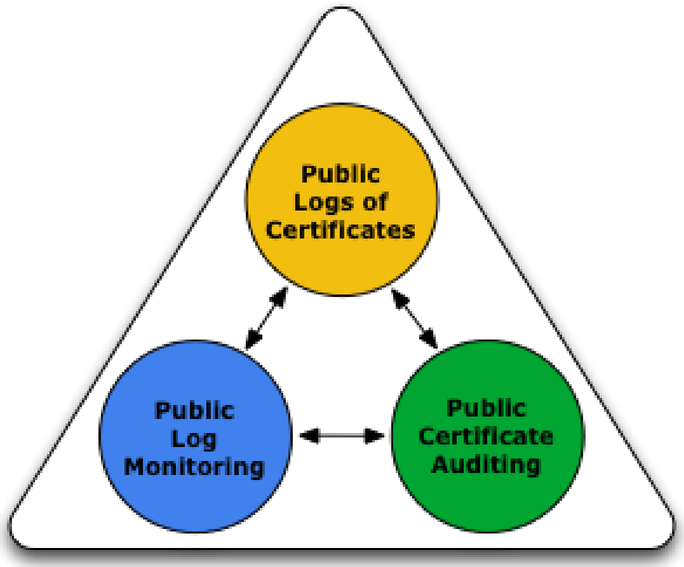
- Network services maintaining cryptographically assured, publicly auditable, append-only records of certificates

Monitors

- Periodically contact all log servers and watch for suspicious certificates

Auditors

- Verify that logs are behaving correctly and are cryptographically consistent
- Check that a particular certificate appears in a log



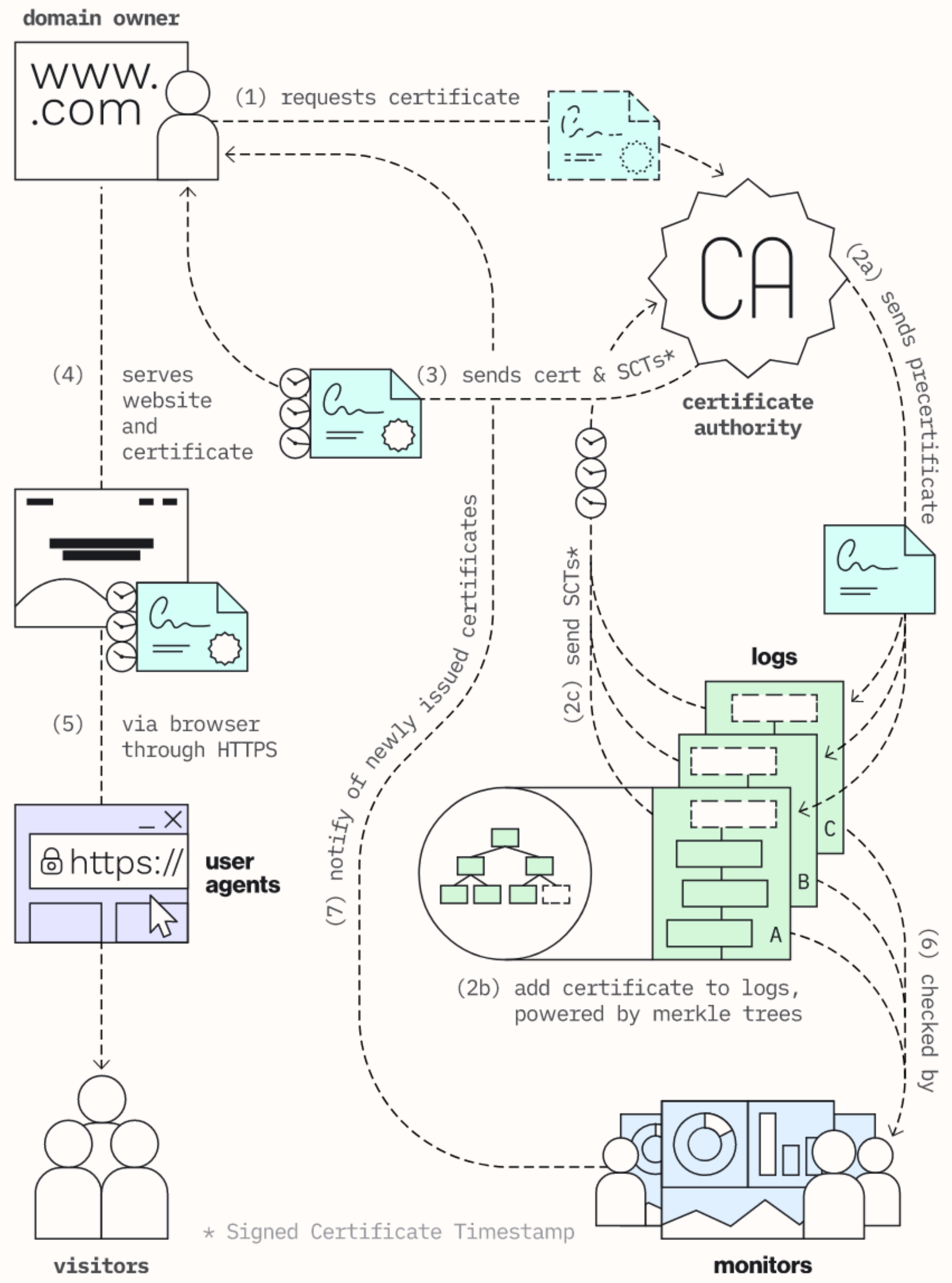
https://certificate.transparency.dev/

Certificates are deposited in public, transparent logs (append-only ledgers)

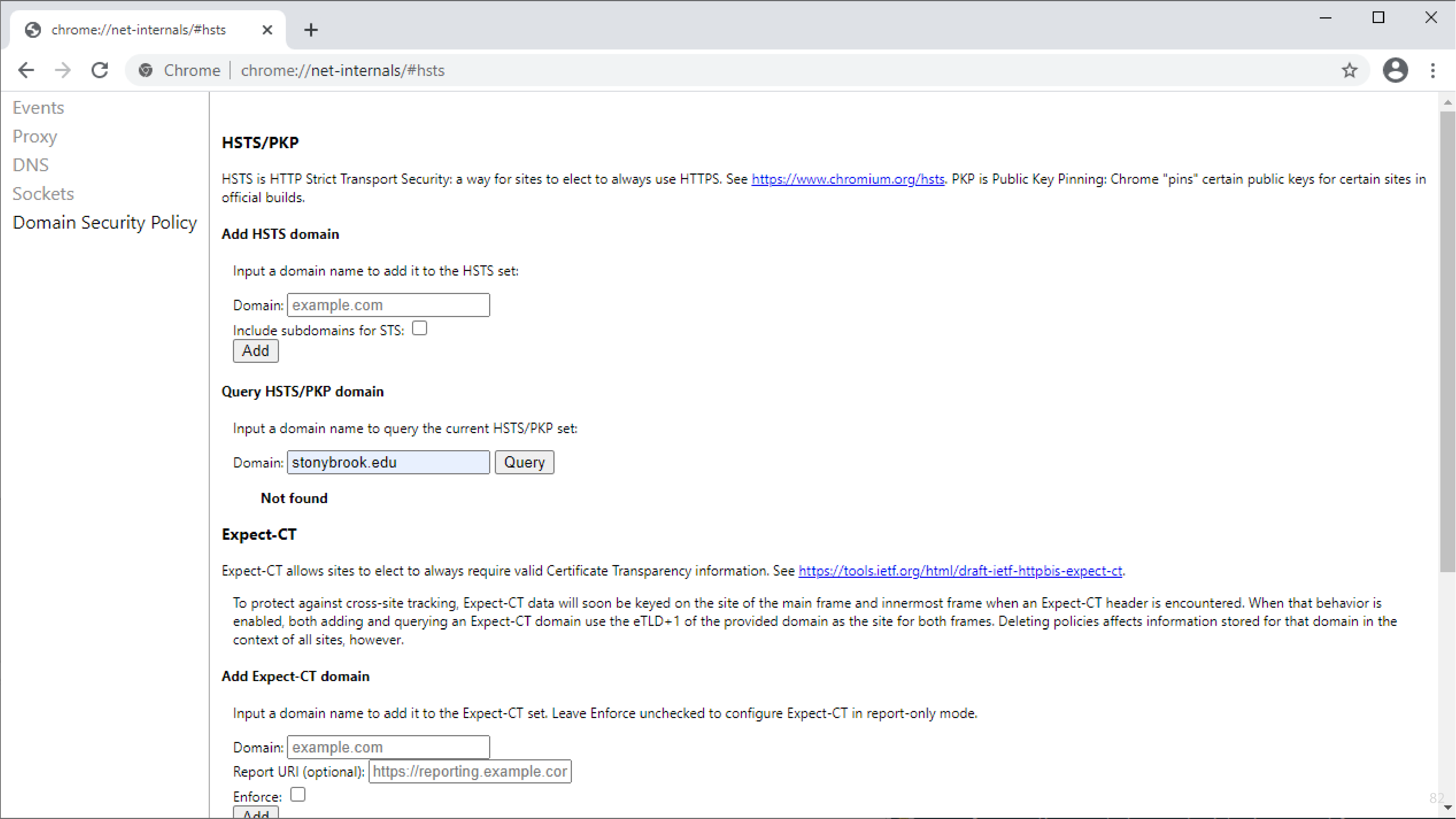
- Distributed and independent: anyone can query them to see what certificates have been included and when
- Append-only: verifiable by Monitors
- Web browsers enforce Certificate Transparency

Logs are cryptographically monitored

- Monitors cryptographically check which certificates have been included in logs
- Domain owners can subscribe to a CT monitor to get updates when precertificates/certificates for those domains are included in any of the logs checked by that monitor



https://certificate.transparency.dev/howctworks/


chrome://net-internals/#hsts
Chrome
chrome://net-internals/#hsts
Events
Proxy
DNS
Sockets
Domain Security Policy
HSTS/PKP
HSTS is HTTP Strict Transport Security: a way for sites to elect to always use HTTPS. See https://www.chromium.org/hsts. PKP is Public Key Pinning: Chrome "pins" certain public keys for certain sites in official builds.
Add HSTS domain
Input a domain name to add it to the HSTS set:
Domain: example.com
Include subdomains for STS:
Add
Query HSTS/PKP domain
Input a domain name to query the current HSTS/PKP set:
Domain: stonybrook.edu
Query
Not found
Expect-CT
Expect-CT allows sites to elect to always require valid Certificate Transparency information. See https://tools.ietf.org/html/draft-ietf-httpbis-expect-ct.
To protect against cross-site tracking, Expect-CT data will soon be keyed on the site of the main frame and innermost frame when an Expect-CT header is encountered. When that behavior is enabled, both adding and querying an Expect-CT domain use the eTLD+1 of the provided domain as the site for both frames. Deleting policies affects information stored for that domain in the context of all sites, however.
Add Expect-CT domain
Input a domain name to add it to the Expect-CT set. Leave Enforce unchecked to configure Expect-CT in report-only mode.
Domain: example.com
Report URI (optional): https://reporting.example.cor
Enforce:

chrome://net-internals/#hsts

Chrome | chrome://net-internals/#hsts

Events
Proxy
DNS
Sockets
Domain Security Policy

HSTS/PKP

HSTS is HTTP Strict Transport Security: a way for sites to elect to always use HTTPS. See https://www.chromium.org/hsts. PKP is Public Key Pinning: Chrome "pins" certain public keys for certain sites in official builds.

Add HSTS domain

Input a domain name to add it to the HSTS set:

Domain: example.com

Include subdomains for STS: ☐

Add

Query HSTS/PKP domain

Input a domain name to query the current HSTS/PKP set:

Domain: google.com Query

Found:
static_sts_domain:
static_upgrade_mode: DEFAULT
static_sts_include_subdomains: false
static_sts_observed: 0
static_pkp_domain: google.com
static_pkp_include_subdomains: true
static_pkp_observed: 1615499290
static_spki_hashes:
sha256/IPMbDAjLVSGntGO3WP53X/zilCVndez5YJ2+vJvhJsA=,sha256/YZPgTZ+woNCCCIW3LH2CxQeLzB/1m42QcCTBSdgayjs=,sha256/hxqRlPTu1bMS/0DITB1SSu0vd4u/8l8TjPgfaAp63Gc=,sha256/Vfd95BwDeSQo+NUYxVEEIlvkOlWY2SalKK1lPhzOx78=,sha256/QXnt2YHvdHR3tJYmQIr0Paosp6t/nggsEGD4QJZ3Q0g=,sha256/mEflZT5enoR1FuXLgYYGqnVEoZvmf9c2bVBpiOjYQ0c=,sha256/iie1VXtL7HzAMF+/PVPR9xzT80kQxdZeJ+zduCB3uj0=
dynamic_sts_domain:
dynamic_upgrade_mode: UNKNOWN
dynamic_sts_include_subdomains:
dynamic_sts_observed:
dynamic_sts_expiry:

Expect-CT