

# Towards a Fully Encrypted Internet

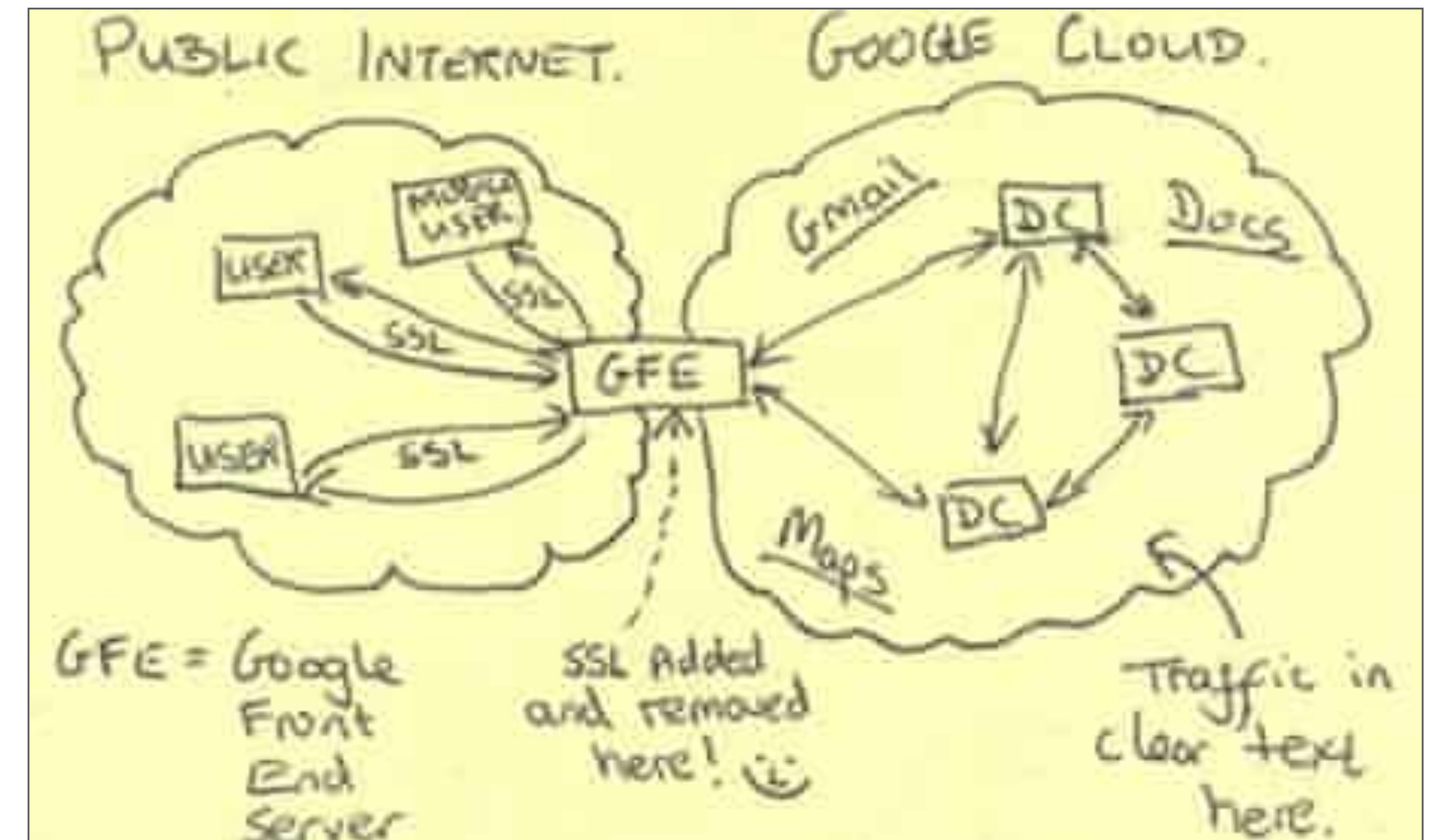
CS244 | Zakir Durumeric

Stanford University

# 2013 Snowden Revelations

Explicit evidence that intelligence agencies are globally wiretapping Internet backbone connections

Massive collection of web traffic, emails, instant messages, contact lists, traffic between cloud providers



# 2014 Heartbleed Vulnerability

Vulnerability in OpenSSL allowed the exposure of the private keys for an estimated 24-55% of the top million most popular websites with HTTPS

Private key leak allowed unencrypting any past traffic for 96% of top million websites



# 2014 State of Encryption

14% of the Alexa Top Million websites supported HTTPS

- Most didn't prefer HTTPS
- Higher adoption than average websites

# 2014 State of Encryption

14% of the Alexa Top Million websites supported HTTPS

- Most didn't prefer HTTPS
- Higher adoption than average websites

Most sites used known-weak versions of TLS

- Only 1 of 4 popular sites supported latest TLS 1.2

4% of websites supported perfect forward secrecy (PFS)

# 2014 State of Encryption

14% of the Alexa Top Million websites supported HTTPS

- Most didn't prefer HTTPS
- Higher adoption than average websites

Most sites used known-weak versions of TLS

- Only 1 of 4 popular sites supported latest TLS 1.2

4% of websites supported perfect forward secrecy (PFS)

Only 1 out of 3 emails were encrypted when sent across the Internet

# Encouraging HTTPS Adoption

**2014:** HTTPS used as a page rank indicator

**Early 2018:** Mozilla announces that new features will require HTTPS

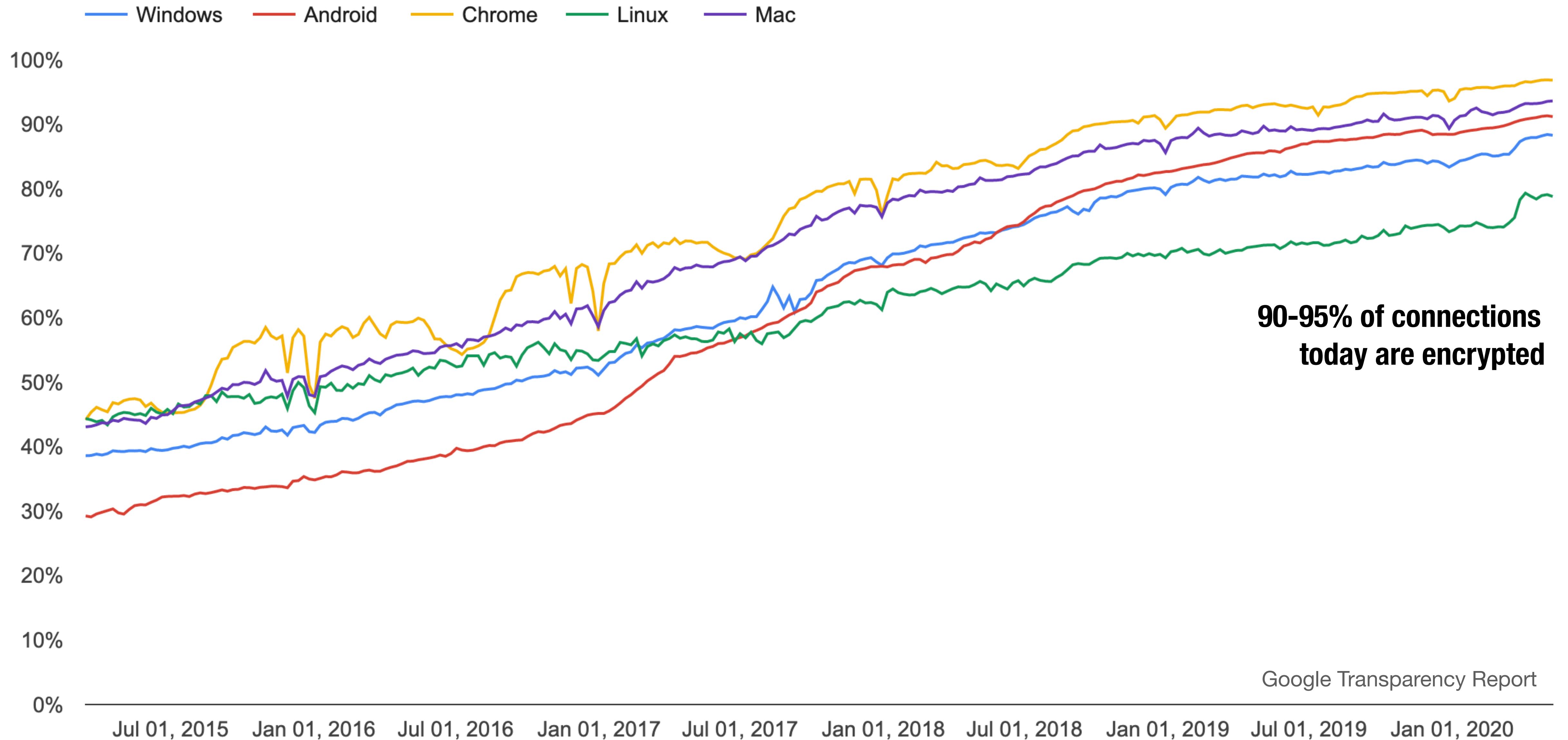
**Late 2018:** New Chrome HTTPS indicators

📘 example.com (HTTPS)

📘 Not secure | example.com (HTTP)

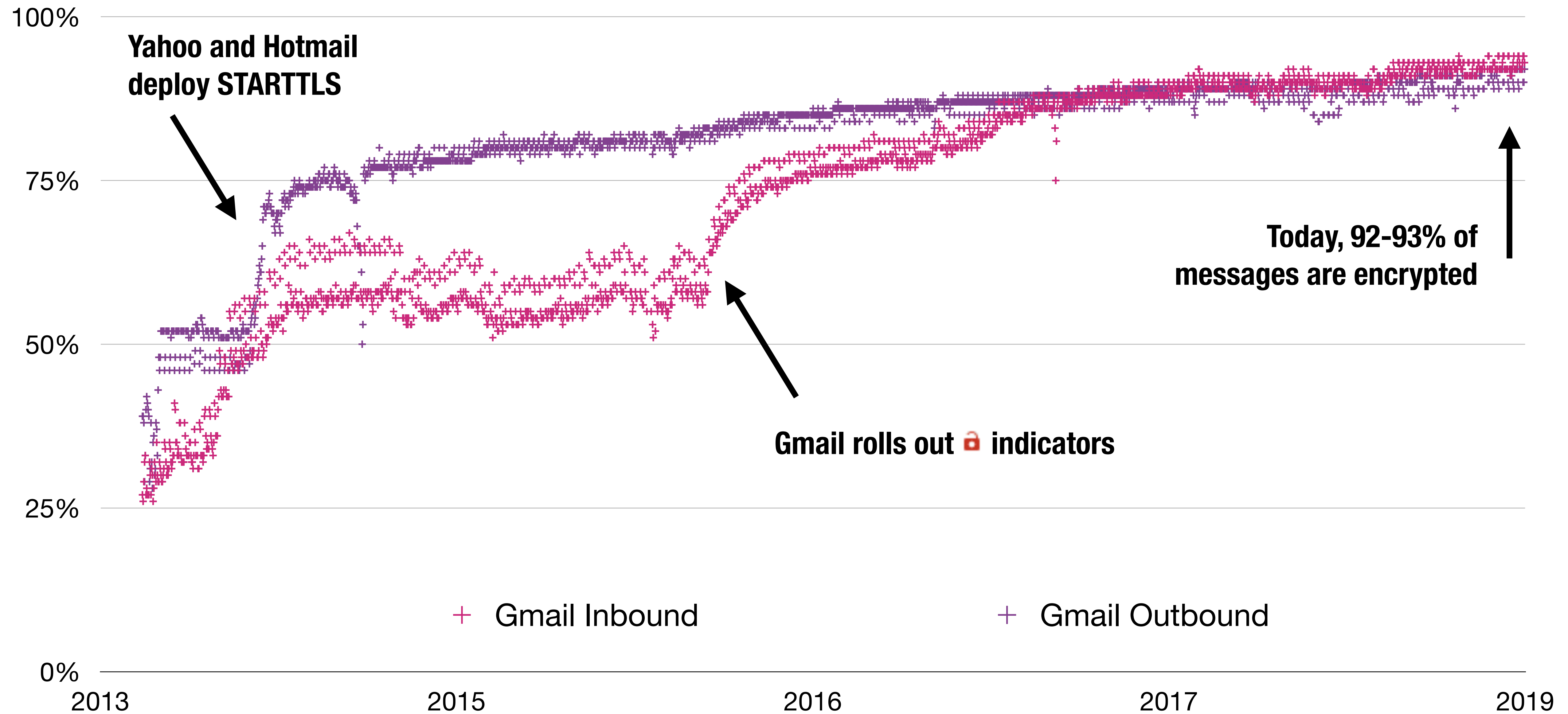


# Chrome Page Loads over HTTPS






# STARTTLS as seen by Gmail



# Timeline of TLS Attacks



A vertical timeline with a central line and horizontal tick marks. Each tick mark is associated with a year on the left and a description of a TLS attack on the right.

2012	BEAST attack against TLS 1.0 CBC ciphers. Many folks recommend using RC4 in response
2012	CRIME attack shows that TLS compression is broken
2013	Lucky 13: padding oracle attack against CBC cipher suites
2014	POODLE Attack: padding oracle attack against SSLv3 results in browsers removing support
2015	FREAK Attack: protocol vulnerability in TLS allows attackers to trick clients into using “export-grade” cryptography if server supports Export Grade RSA
2015	Logjam Attack: protocol vulnerability found that enables attackers to downgrade some connections to export grade Diffie-Hellman. Browsers remove traditional D-H support.
2016	RC4 deprecation: after a string of attacks against RC4, major browsers remove support
2016	DROWN attack: cross-protocol attack on export-grade AES
2016	Sweet32: Birthday attacks on 64-bit block ciphers like 3DES
2017	First public SHA-1 collision

# Timeline of TLS Attacks

2012	BEAST attack against TLS 1.0 CBC ciphers. Many folks recommend using RC4 in response
2012	CRIME attack shows that TLS compression is broken
2013	Lucky 13: padding oracle attack against CBC cipher suites
2014	POODLE Attack: padding oracle attack against SSLv3 results in browsers removing support
2015	FREAK Attack: protocol vulnerability in TLS allows attackers to trick clients into using “export-grade” cryptography if server supports Export Grade RSA
2015	Logjam Attack: protocol vulnerability found that enables attackers to downgrade some connections to export grade Diffie-Hellman. Browsers remove traditional D-H support.
2016	RC4 deprecation: after a string of attacks against RC4, major browsers remove support
2016	DROWN attack: cross-protocol attack on export-grade AES
2016	Sweet32: Birthday attacks on 64-bit block ciphers like 3DES
2017	First public SHA-1 collision

# Timeline of TLS Attacks

2012	BEAST attack against TLS 1.0 CBC ciphers. Many folks recommend using RC4 in response
2012	CRIME attack shows that TLS compression is broken
2013	Lucky 13: padding oracle attack against CBC cipher suites
2014	POODLE Attack: padding oracle attack against SSLv3 results in browsers removing support
2015	FREAK Attack: protocol vulnerability in TLS allows attackers to trick clients into using “export-grade” cryptography if server supports Export Grade RSA
2015	Logjam Attack: protocol vulnerability found that enables attackers to downgrade some connections to export grade Diffie-Hellman. Browsers remove traditional D-H support.
2016	RC4 deprecation: after a string of attacks against RC4, major browsers remove support
2016	DROWN attack: cross-protocol attack on export-grade AES
2016	Sweet32: Birthday attacks on 64-bit block ciphers like 3DES
2017	First public SHA-1 collision

**Full Timeline:** <https://www.feistyduck.com/ssl-tls-and-pki-history/>

# Timeline of TLS Attacks

2012	BEAST attack against TLS 1.0 CBC ciphers. Many folks recommend using RC4 in response
2012	CRIME attack shows that TLS compression is broken
2013	Lucky 13: padding oracle attack against CBC cipher suites
2014	POODLE Attack: padding oracle attack against SSLv3 results in browsers removing support
2015	FREAK Attack: protocol vulnerability in TLS allows attackers to trick clients into using “export-grade” cryptography if server supports Export Grade RSA
2015	Logjam Attack: protocol vulnerability found that enables attackers to downgrade some connections to export grade Diffie-Hellman. Browsers remove traditional D-H support.
2016	RC4 deprecation: after a string of attacks against RC4, major browsers remove support
2016	DROWN attack: cross-protocol attack on export-grade AES
2016	Sweet32: Birthday attacks on 64-bit block ciphers like 3DES
2017	First public SHA-1 collision

# U.S. Export-Grade Cryptography

Until 1992, the United States severely restricted what cryptographic technology could be exported outside of the country. Loosened slightly.

Early 1990s: Two versions of Netscape Browser — US version had full strength crypto (e.g., 1024-bit RSA, 128-bit RC4) and Export version (40-bit RC2, 512-bit RSA)

1996: Bernstein v. the United States: Ninth Circuit Court of Appeals ruled that software source code was speech protected by the First Amendment and that the government's regulations preventing its publication were unconstitutional

Decision later withdrawn, but U.S. changed policy to allow, no precedent set

# Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice

David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J .  
Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta,  
Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Beguelin, and Paul Zimmermann

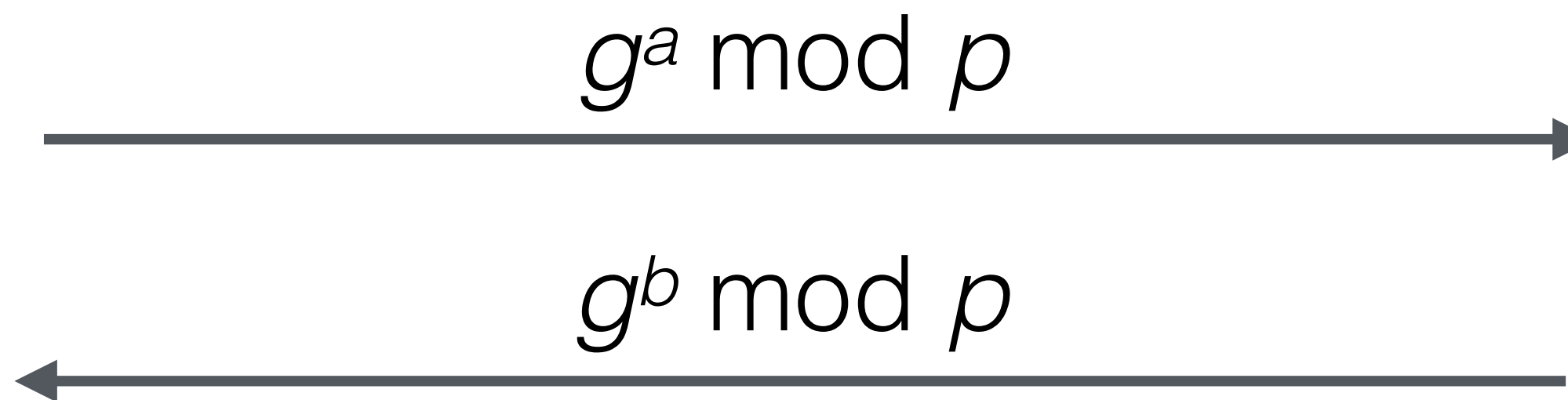


# Diffie-Hellman Key Exchange

First published key exchange algorithm

## Public Parameters

- $p$  (a large prime)
- $g$  (generator for group  $p$ )



$$g^{ab} \bmod p == g^{ba} \bmod p$$

# Diffie-Hellman on the Internet

Diffie-Hellman is pervasive on the Internet today

## **Primary Key Exchange**

- SSH
- IPSEC VPNs

## **Ephemeral Key Exchange**

- HTTPS
- SMTP, IMAP, POP3
- all other protocols that use TLS

“Sites that use perfect forward secrecy can provide better security to users in cases where the encrypted data is being monitored and recorded by a third party.”

“Ideally the DH group would match or exceed the RSA key size but 1024-bit DHE is arguably better than straight 2048-bit RSA so you can get away with that if you want to.”

“With Perfect Forward Secrecy, anyone possessing the private key and a wiretap of Internet activity can decrypt nothing.”

# 2015 Diffie-Hellman Support

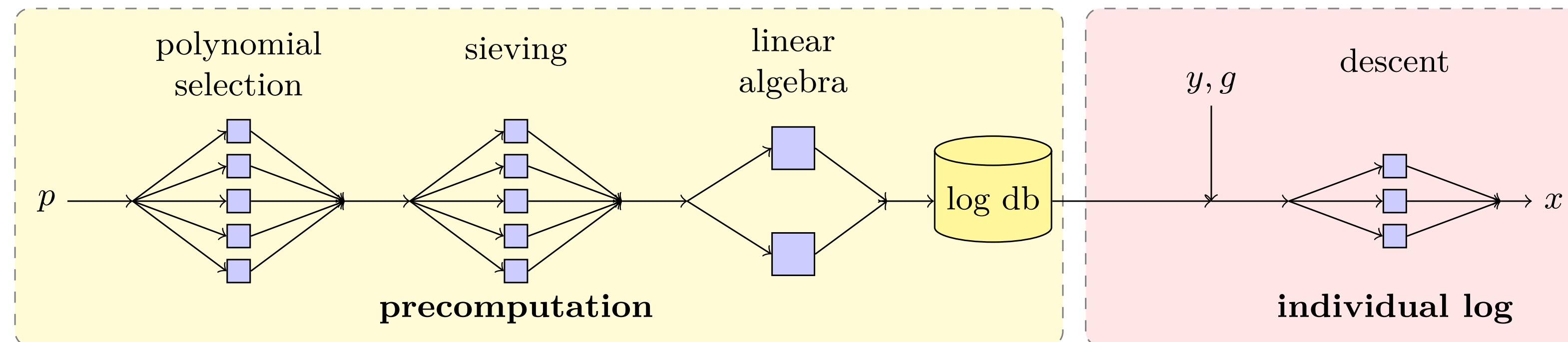
Protocol	Support
HTTPS (Top Million Websites)	68%
HTTPS (IPv4, Browser Trusted)	24%
SMTP + STARTTLS	41%
IMAPS	75%
POP3S	75%
SSH	100%
IPSec VPNs	100%

# Breaking Diffie-Hellman

Computing discrete log is best known attack against DH

In other words, Given  $g^x \equiv y \pmod{p}$ , compute  $x$

## Number Field Sieve

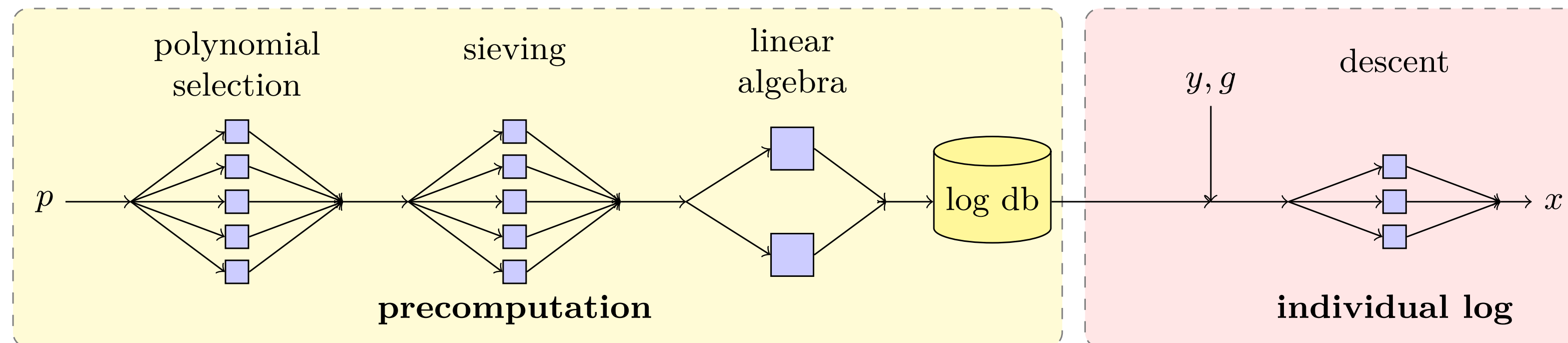


# Breaking Diffie-Hellman

Computing discrete log is best known attack against DH

In other words, Given  $g^x \equiv y \pmod{p}$ , compute  $x$

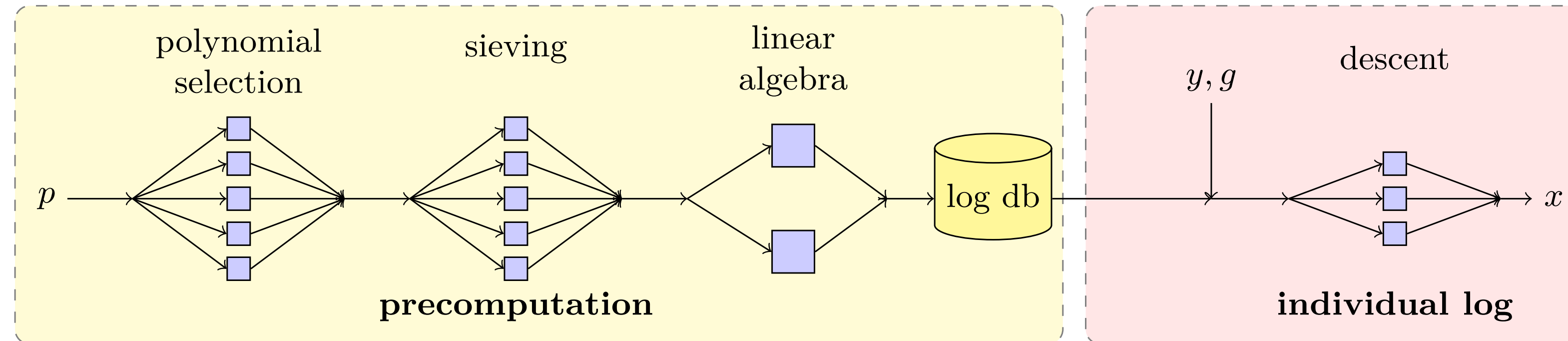
## Number Field Sieve



**Pre-computation is only dependent on  $p$ !**

# Breaking Diffie-Hellman

## Number Field Sieve



	Sieving	Linear Algebra	Descent
DH-512	2.5 core years	7.7 core years	10 core min.



# Lost in Translation

This was known within the cryptographic community

However, not within the systems community

66% of IPSec VPNs use a single 1024-bit prime

# Lost in Translation

This was known within the cryptographic community

However, not within the systems community

66% of IPSec VPNs use a single 1024-bit prime

**Are the groups used in practice still  
secure given this “new” information?**

# 512-bit Keys and the Logjam Attack on TLS

# Diffie-Hellman in TLS

The majority of HTTPS websites use 1024-bit DH keys

However, nearly 8.5% of Top 1M still support *Export DHE*

Source	Popularity
Apache	82%
mod_ssl	10%
Other (463 distinct primes)	8%

# Normal TLS Handshake

client hello: client random, ciphers (... DHE ...)

server hello: server random, chosen cipher



# Normal TLS Handshake

client hello: client random, ciphers (... DHE ...)

server hello: server random, chosen cipher

certificate,  $p$ ,  $g$ ,  $g^a$ ,  $\text{Sign}_{\text{CertKey}}(p, g, g^a)$

$g^b$

$K_{\text{ms}}: \text{KDF}(g^{ab}, \text{client random}, \text{server random})$



# Normal TLS Handshake

client hello: client random, ciphers (... DHE ...)

server hello: server random, chosen cipher

certificate,  $p$ ,  $g$ ,  $g^a$ ,  $\text{Sign}_{\text{CertKey}}(p, g, g^a)$

$g^b$

$K_{\text{ms}}: \text{KDF}(g^{ab}, \text{client random}, \text{server random})$

client finished:  $\text{Sign}_{K_{\text{ms}}}(\text{Hash}(m1 \mid m2 \mid \dots))$

server finished:  $\text{Sign}_{K_{\text{ms}}}(\text{Hash}(m1 \mid m2 \mid \dots))$

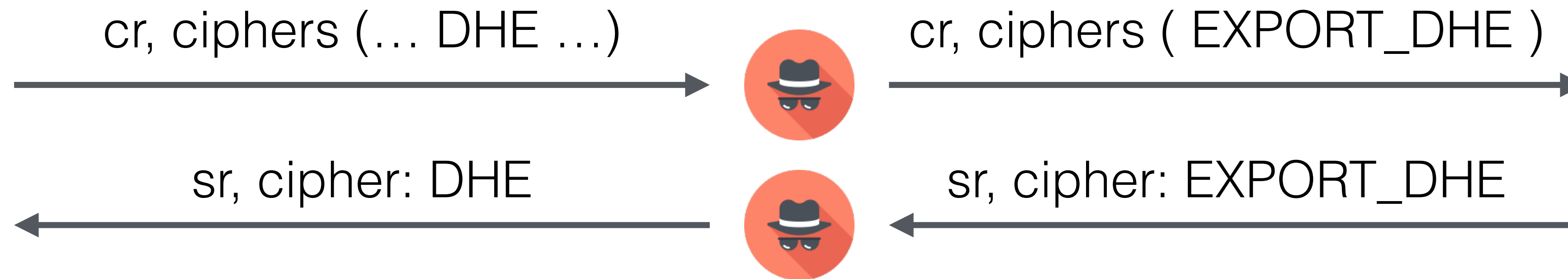




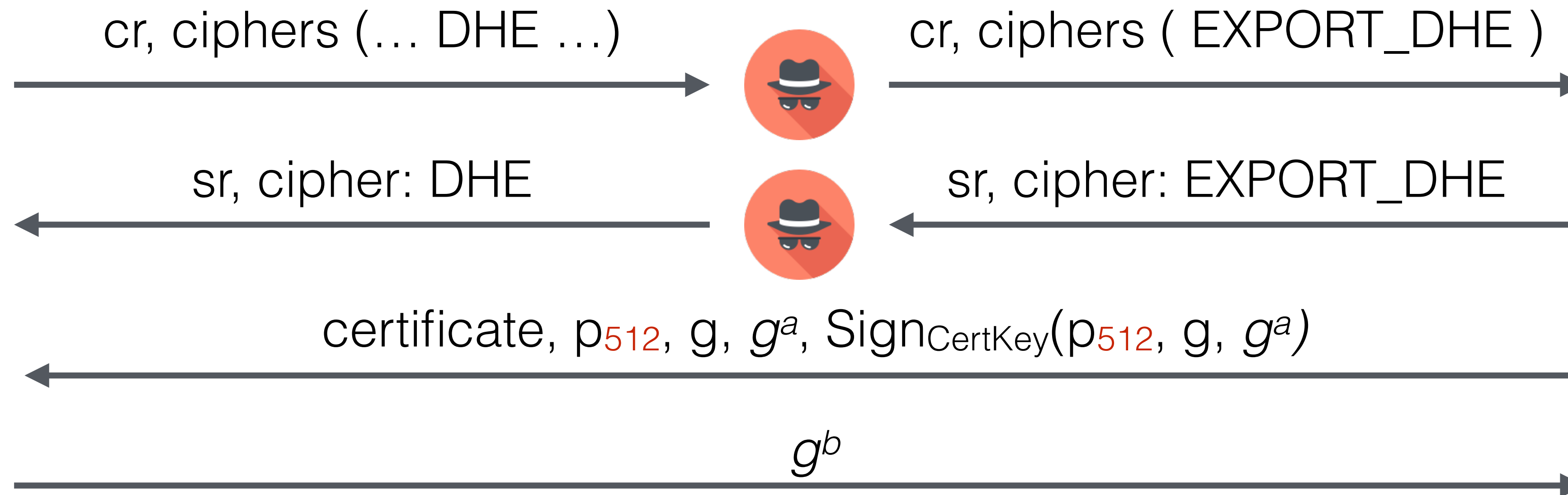
# Logjam Attack



# Logjam Attack

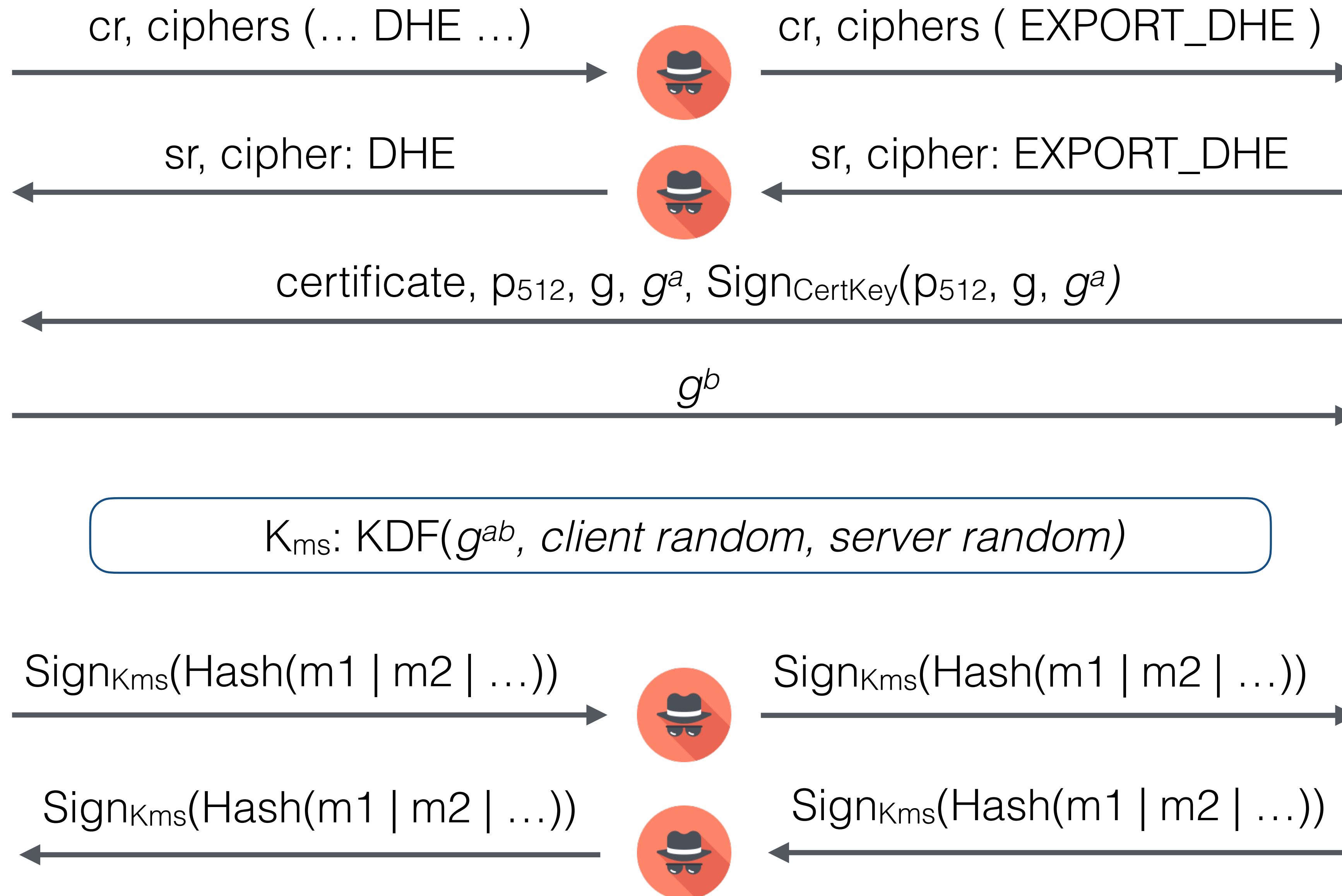


# Logjam Attack



$K_{ms}: \text{KDF}(g^{ab}, \text{client random}, \text{server random})$

# Logjam Attack



# Computing 512-bit Discrete Logs

We modified CADO-NFS to compute two common primes

1 week pre-computation, individual log ~70 seconds

	polysel	sieving	linalg	descent
	2000-3000	cores	288	cores
DH-512	3 hours	15 hours	120 hours	70 seconds

# Logjam Mitigation

## Browsers

- have raised minimum size to 768-bits
- ~~plan to move to 1024 bit in the future~~
- plan to drop all support for DHE

## Server Operators

- Disable export ciphers!!
- ~~Use a 2048 bit or larger DHE key~~
- ~~If stuck using 1024 bit, generate a unique prime~~
- Moving to ECDHE

# 768- and 1024-bit Keys



# Breaking One 1024-bit DH Key

Estimation process is convoluted due to the number of parameters that can be tuned.

Crude estimations based on asymptotic complexity:

	Sieving core-years	Linear Algebra core-years	Descent core-time
RSA-512	0.5	0.33	
DH-512	2.5	7.7	10 mins
RSA-768	800	100	
DH-768	8,000	28,500	2 days
RSA-1024	1,000,000	120,000	
DH-1024	10,000,000	35,000,000	30 days

# Custom Hardware

If you went down this route, you would build ASICs

Prior work from Geiselmann and Steinwandt (2007) estimates ~80x speed up from custom hardware.

≈\$100Ms of HW precomputes one 1024-bit prime/year

# Custom Hardware

If you went down this route, you would build ASICs

Prior work from Geiselmann and Steinwandt (2007) estimates ~80x speed up from custom hardware.

≈\$100Ms of HW precomputes one 1024-bit prime/year

## **For context... annual budgets for the U.S.**

- Consolidated Cryptographic Program: 10.5B
- Cryptanalytic IT Services: 247M
- Cryptanalytic and exploitation services: 360M

# TLS 1.3

# TLS 1.3 What's New?

## Removed:

- Problematic features from the past like compression, renegotiation
- Known broken ciphers like MD-5, SHA-1, RC4, 3DES, CBC mode, traditional finite-field Diffie-Hellman, export ciphers, user defined groups
- Non-PFS (perfect forward secret) handshakes

## Added:

- + Simplified handshake with one fewer round trip
- + Protection against downgrade attacks (e.g., signature over entire exchange)
- + Support for newer elliptic curves (e.g., x25519 and 448)
- + Zero RTT Session Resumption (performance win)

# TLS 1.3 Design

TLS 1.3 was finalized in 2018! Process took ~5 years.

One of first major protocols to involve academic community during design.  
Uncovered multiple attacks, including a downgrade, cross-protocol, and key-sharing attack

Empirical tests helped design a handshake that minimizes interference with broken middle boxes

# Web PKI

# Web PKI

**2010:** No visibility into who was trusted to issue certificates

**2013:** We find that ~700 organizations controlled certificates through large-scale scans of web servers. No assured visibility into certificates—only know if we stumbled upon the cert in the wild

~10-20% of certificate were constructed incorrectly

Example: Turktrust, mis-issued to a certificate in 2012 to a bus station that was capable of signing browser trusted certificates for every website.

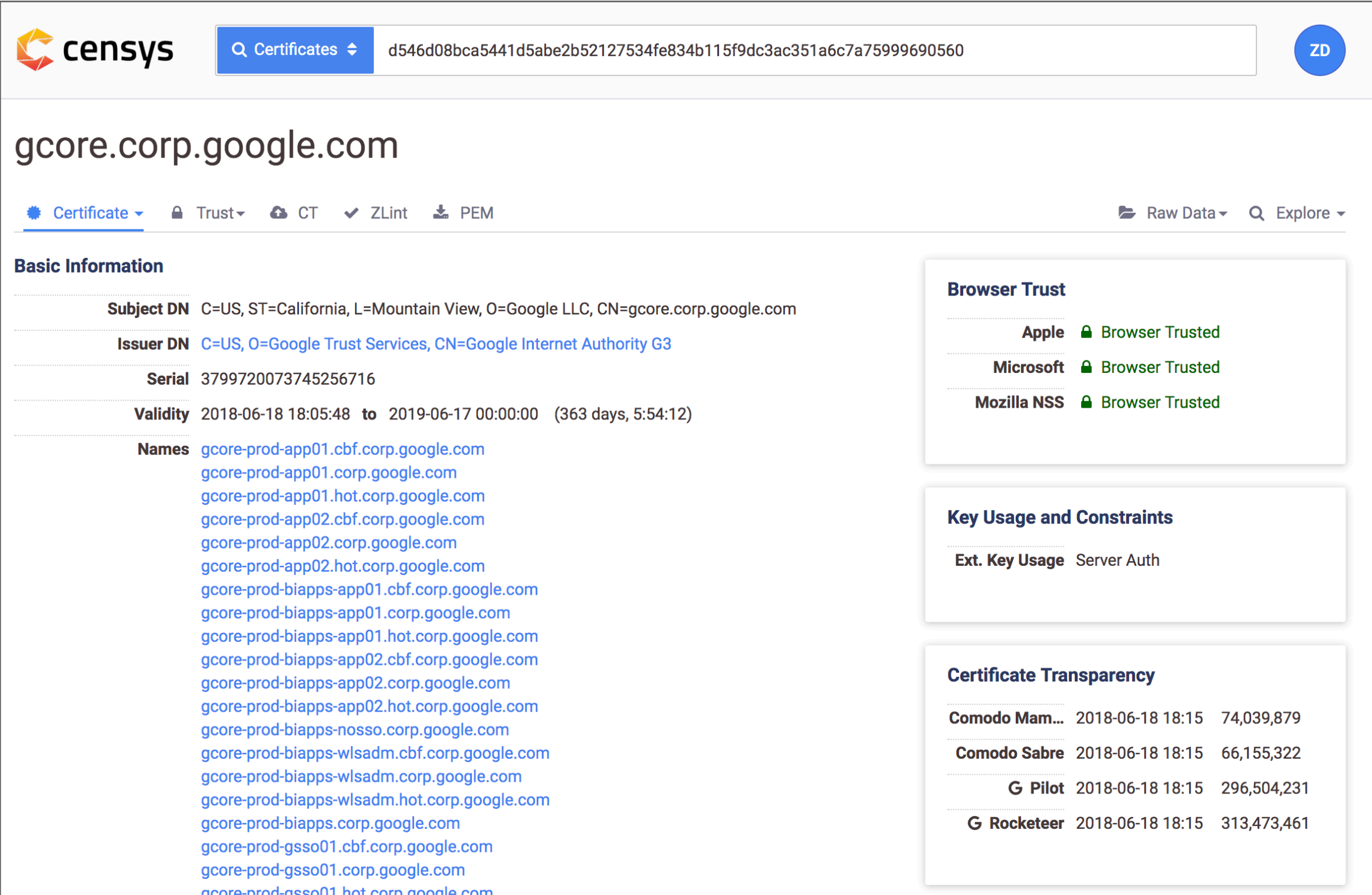


# Certificate Transparency

Chrome, Firefox, Safari require browser trusted certificates to be present in Certificate Transparency logs

Enabled real-time monitoring of new certificates for problems.

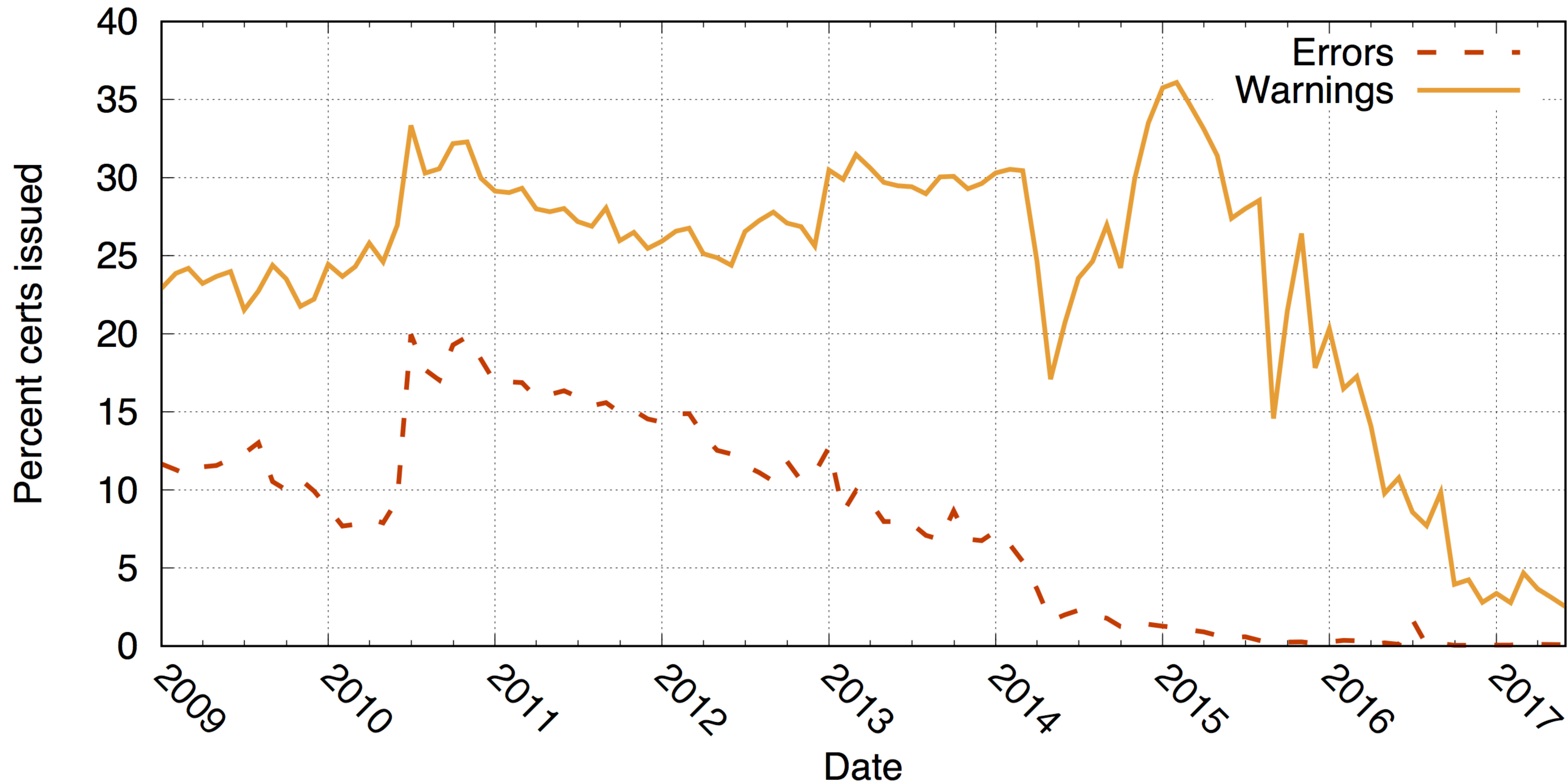
Chrome and Firefox have removed several problematic authorities

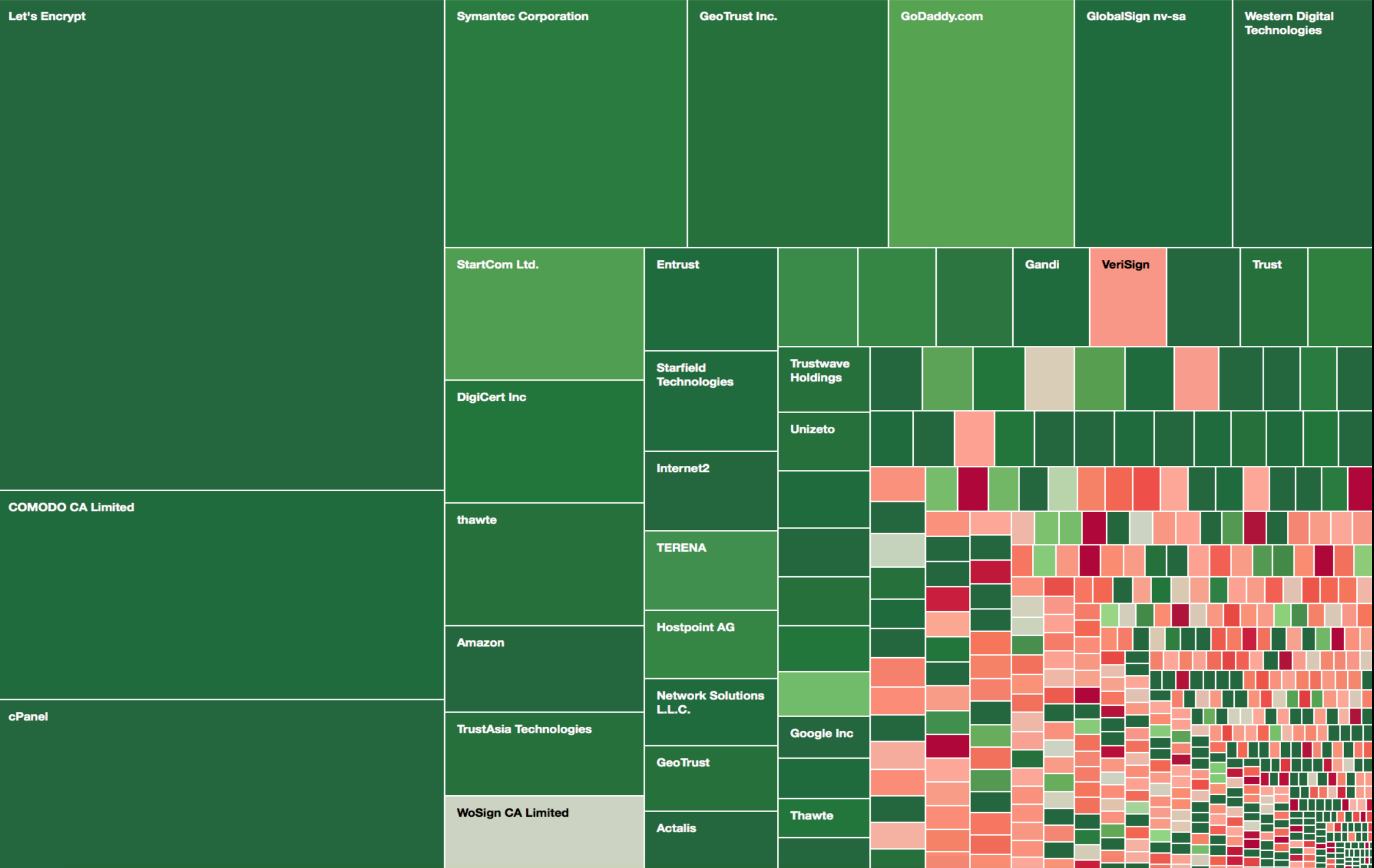


The screenshot displays the Censys web interface for a specific certificate. The header shows the Censys logo, a search bar with the text "Certificates", and a search ID "d546d08bca5441d5abe2b52127534fe834b115f9dc3ac351a6c7a75999690560". The main heading is "gcore.corp.google.com". Below this, there are tabs for "Certificate", "Trust", "CT", "ZLint", and "PEM". The "Certificate" tab is active, showing "Basic Information" and "Browser Trust" sections. The "Basic Information" section includes fields for Subject DN, Issuer DN, Serial, Validity, and Names. The "Browser Trust" section shows the certificate is trusted by Apple, Microsoft, and Mozilla NSS. The "Key Usage and Constraints" section shows the extended key usage as "Server Auth". The "Certificate Transparency" section shows a list of certificates issued by Comodo Mammoth, Comodo Sabre, G Pilot, and G Rocketeer.

Certificate Transparency			
Comodo Mammoth	2018-06-18 18:15	74,039,879	
Comodo Sabre	2018-06-18 18:15	66,155,322	
G Pilot	2018-06-18 18:15	296,504,231	
G Rocketeer	2018-06-18 18:15	313,473,461	

Search crt.sh or censys.io for certificates

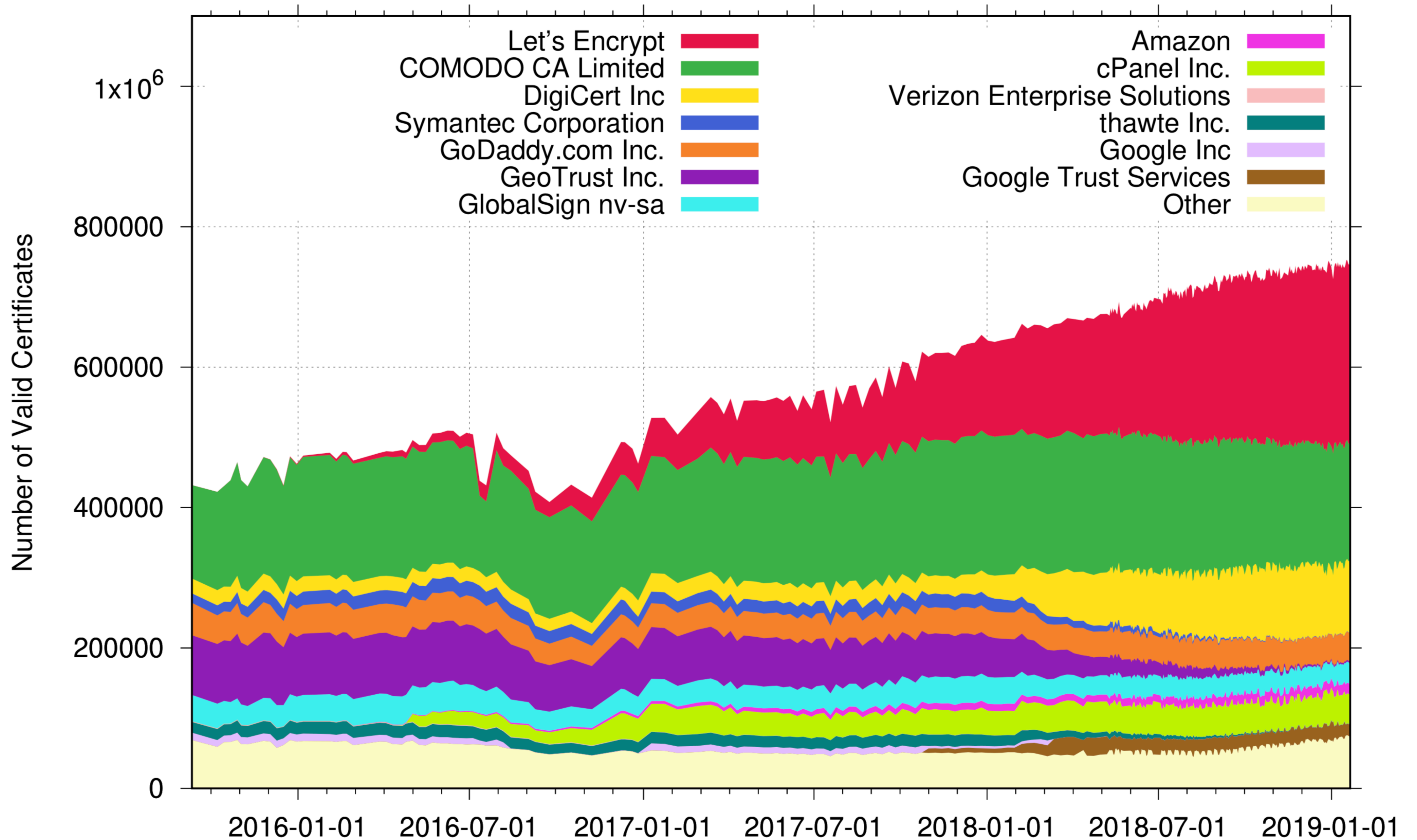




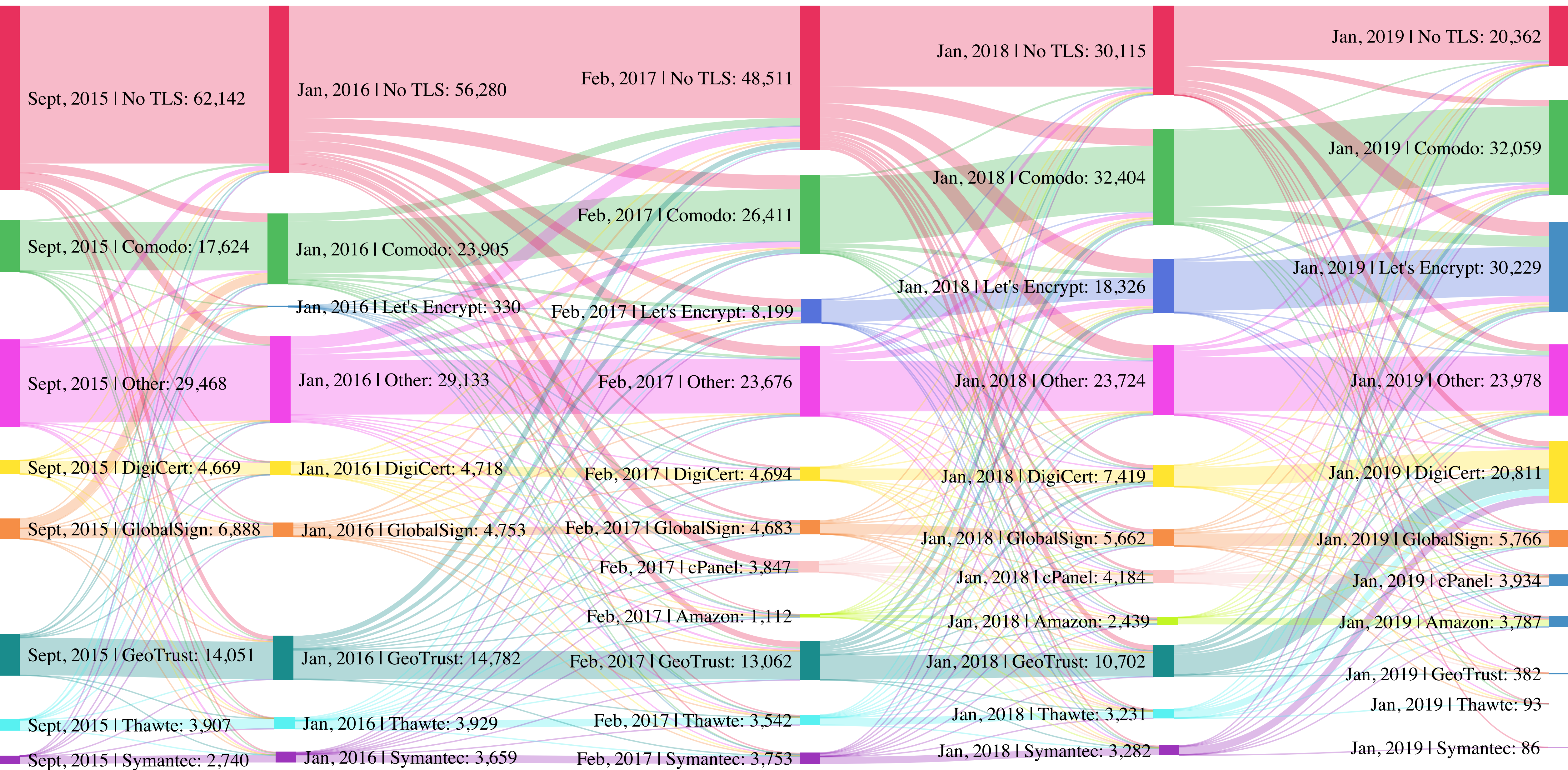




# Let's Encrypt







CA Market Share 2015 -> 2019 Alexa Top Million Websites