How mature is your HTTPS implementation ?

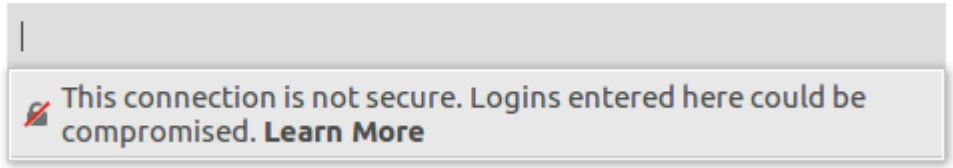
By Renaud Dubois

Some of the latest news about HTTPS

01/12/2017

Introduction : Why https ?

- Higher security & privacy than HTTP
  - Specially for sensitive data



- Better Google ranking

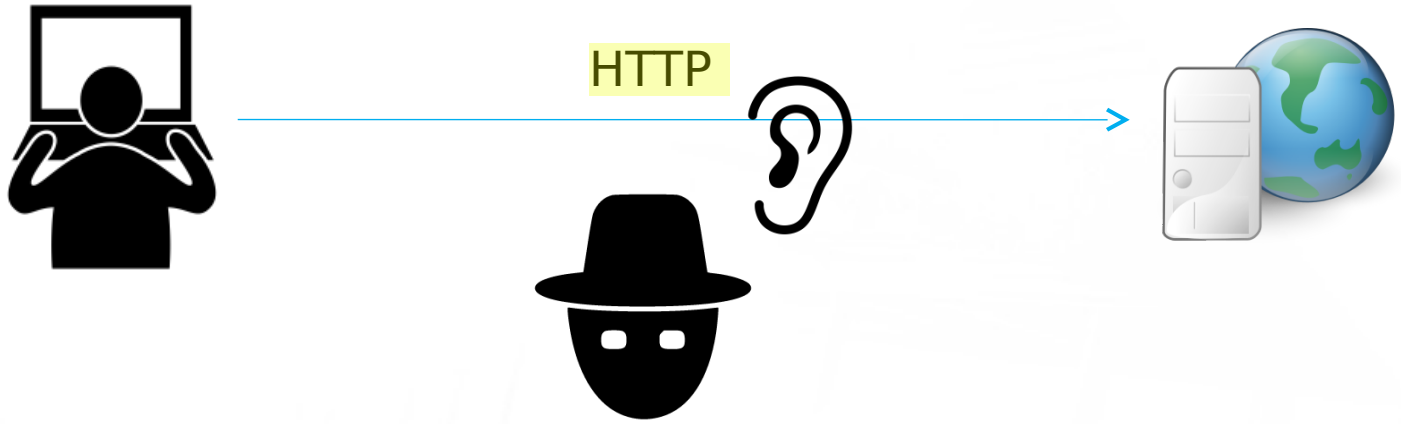
- Follow the initiative to make the web safer (initiatives such as Let’s encrypt, HTTPS everywhere)

HTTP only: main risks

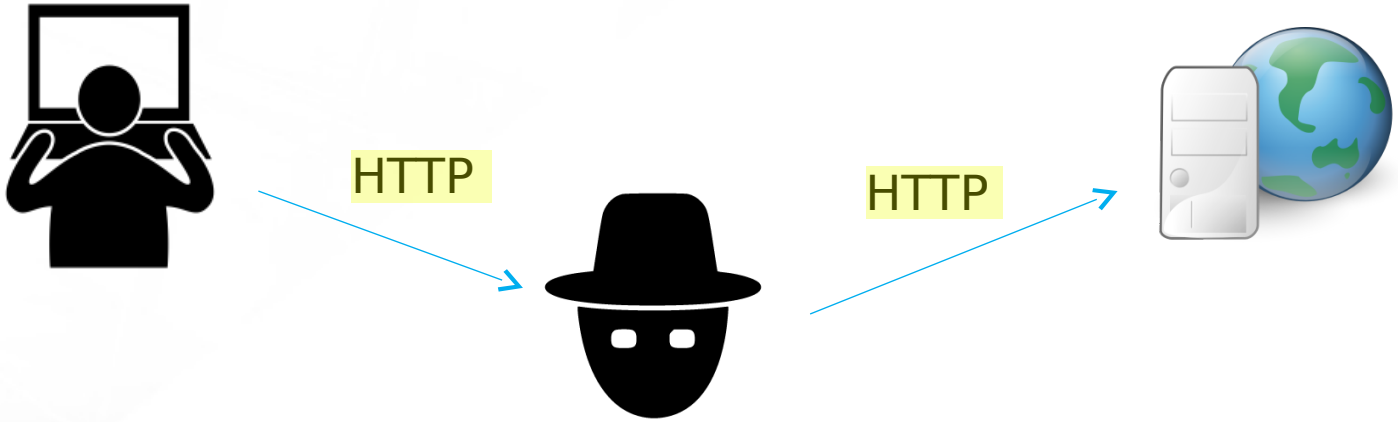
- Confidentiality
  - Credentials eavesdropping (login/password, cookies,..)
  - Data eavesdropping
- Integrity
  - Data manipulation (injection – replacement) including on files downloaded
  - Dynamic code injection (Javascript)
  - ...

HTTP threats: let's see in practice (1/3)

- Passive spoofing/eavesdropping with a Rogue Access WiFi Point
- Passive spoofing from a network or telecom equipment

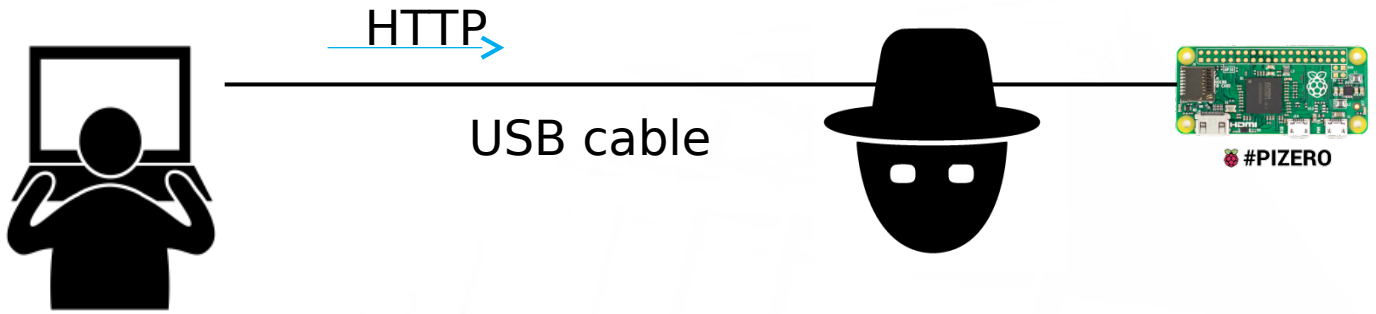


- Man-in-the-middle (e.g. based ARP poisoning in IPv4; fake RA in IPv6)



HTTP threats : let's see in practice (2/3)

- Cookie based credentials hijack (e.g. via PoisonTap and Raspberry Pi Zero)



- Emulate Ethernet device over USB
- Run DHCP, DNS Server
- Hijack all internet traffic
- Allow leaking over HTTP request and catching user's cookie
  - Force HTTP traffic (even for HTTPS website)
  - Grab the users' cookie (if the website runs without HSTS or if 'Secure flag' is not enabled on the cookies)

HTTP threats

What can a bad guy concretely do

Injecting content in the html pages

Redirect to a phishing website

Stealing login/password

Stealing existing session (cookie)

Replacing downloaded files (by malware)

HTTPS implementation 1/2

Partial HTTPS implementation limited to the login page (year ‘90)

HTTPS GET /authentication

HTTP GET /content

Secure architecture ??

→ Insufficient and still unsecure

→ Eavesdropping still possible of the session after authentication with the HTTP content (cookie)

HTTPS implementation 2/2

Full HTTPS implementation

HTTPS GET
/authentication
HTTPS GET /content

→ Mitigate passive spoofing

Secure architecture ??

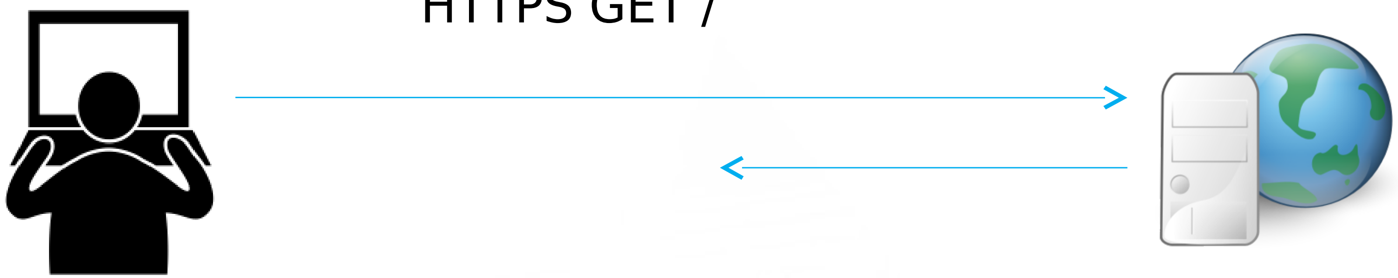
→ Doesn’t always mitigate MITM attack ␣ Downgrade attack to HTTP often still possible in some cases

HTTPS implementation demo let’s see in practice

Demo

HTTPS with HSTS

HTTPS GET /



HTTPS: Strict-Transport-Security: max-age=<in seconds>

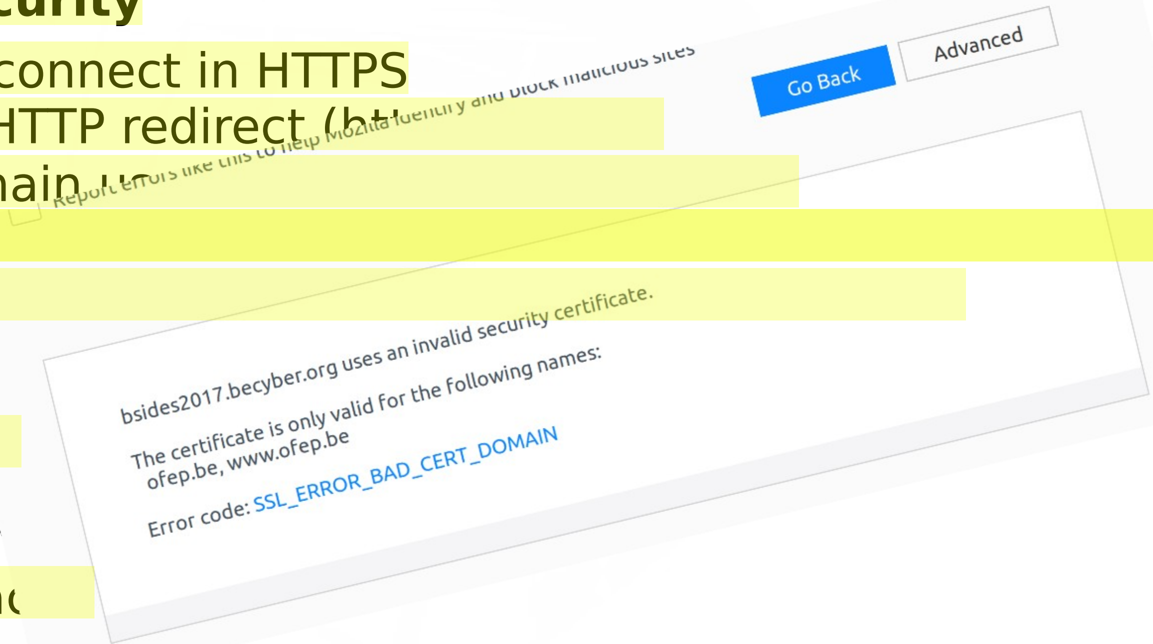
HTTP Strict Transport Security

- Force the browser to always connect in HTTPS
- To use in combination with a HTTP redirect (ht
  - Future request to the domain us
  - In case the connection … wser and the site
  - Stay inaccessible / The user cannot “bypa
    Exception”)

→ Mitigate passive spoofir

→ Mitigate cookie based cr

→ Mitigate some MITM attac



HTTPS with HSTS: in details

For specific domains/subdomain or for all subdomains (*.mydomain.com)
Good practice: implement HSTS for www.domain.com and domain.com

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

Setting *includeSubDomains* on www.mydomain.com also applied for subdomains (e.g. app1.mydomain.com)
Be careful: could impact sites on subdomain that are not yet HTTP enabled

Preloaded list available in the browsers (Chrome, Firefox, Opera, Safari, IE 11 and Edge)
https://hstspreload.org/
- Mitigate the possible attack on the first connection and the time based attacks

HTTPS with HSTS: in details

Considered as « HIGH » security benefit by the Web Security Mozilla Sheet
Recommended « max-age » final value: 2 years (63072000 seconds)

How to still MITM websites using HSTS not part of the preload list ?

- First connection remains unprotected (with a risk of a downgrade attack and stripping the HSTS header)
- Vulnerable to time based attacks (e.g. false NTP packet)

Privacy:
“Supercookie” could lead to privacy issues

<img src=http://a.mydomain.com/pic.jpg »> _ required HTTPS in future = Y
<img src=http://b.mydomain.com/pic.jpg »> _ required HTTPS in future = N
<img src=http://c.mydomain.com/pic.jpg »> _ required HTTPS in future = Y
<img src=http://d.mydomain.com/pic.jpg »> _ required HTTPS in future = Y

HTTPS with HSTS: incognito mode

- HSTS is supported by all the recent versions of browser (incl. IE on Win 7 with KB3058515)
- Status of the browser and HSTS « Normal mode » vs « Incognito/Private mode »

- Privacy vs Security

Browser	Shared between normal & private mode
Firefox 56	No
Internet Explorer 11 (KB3058515)	No
Chrome 61	Yes
Safari 11	Yes

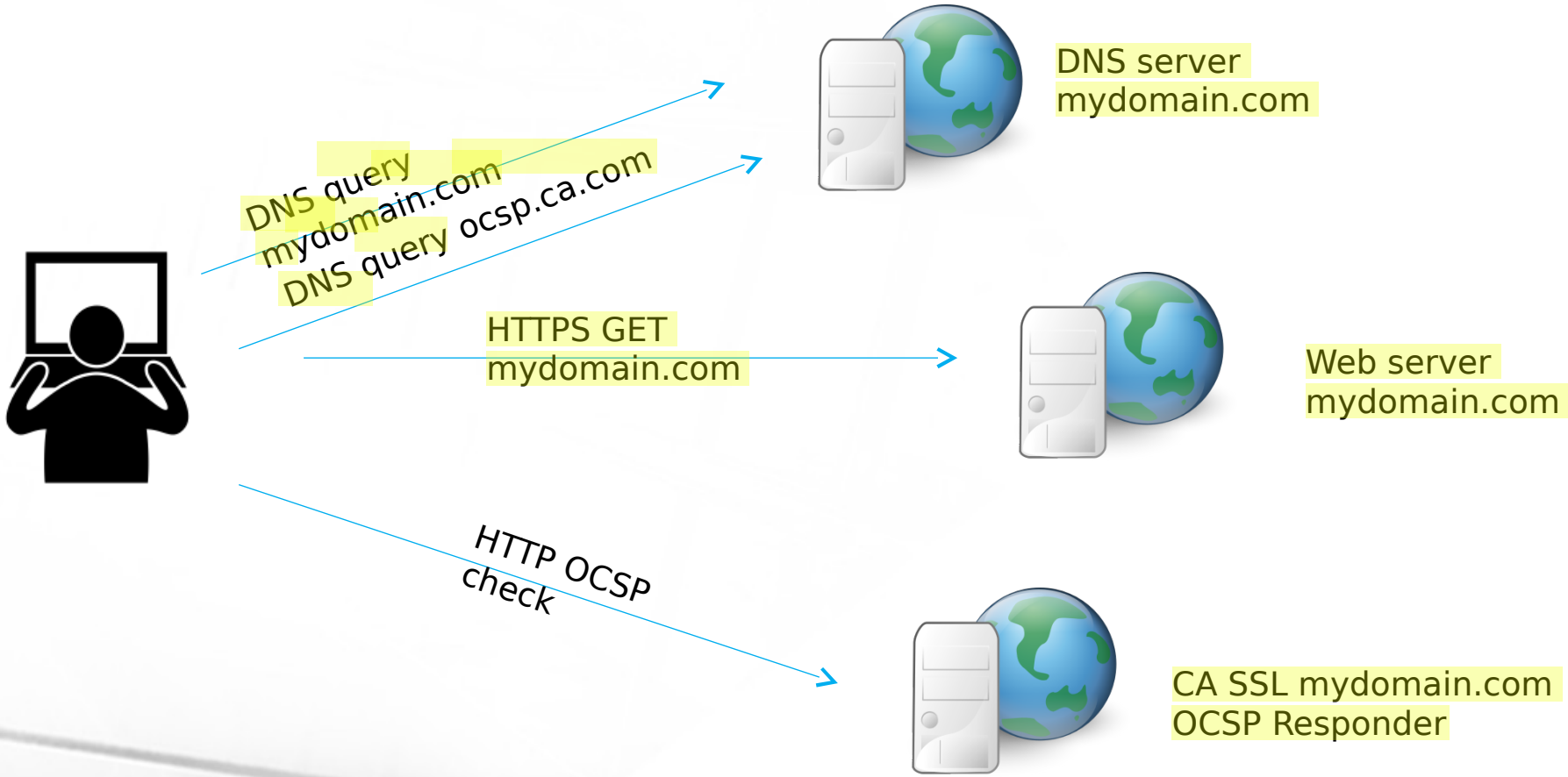
Browser	Shared between 2 private mode sessions
Firefox 56	Yes
Internet Explorer 11 (KB3058515)	No
Chrome 61	Yes

OCSP: Introduction

- Client must verify the validity of the server certificate
  - CRL ␣ huge list ␣ latency to download
  - OCSP (Online Certificate Status Protocol) ␣ more lightweight
    - ␣ extra OCSP request to a 3d party OCSP responder

OCSP: Presentation

- Regular OCSP browser validation



OCSP: Presentation

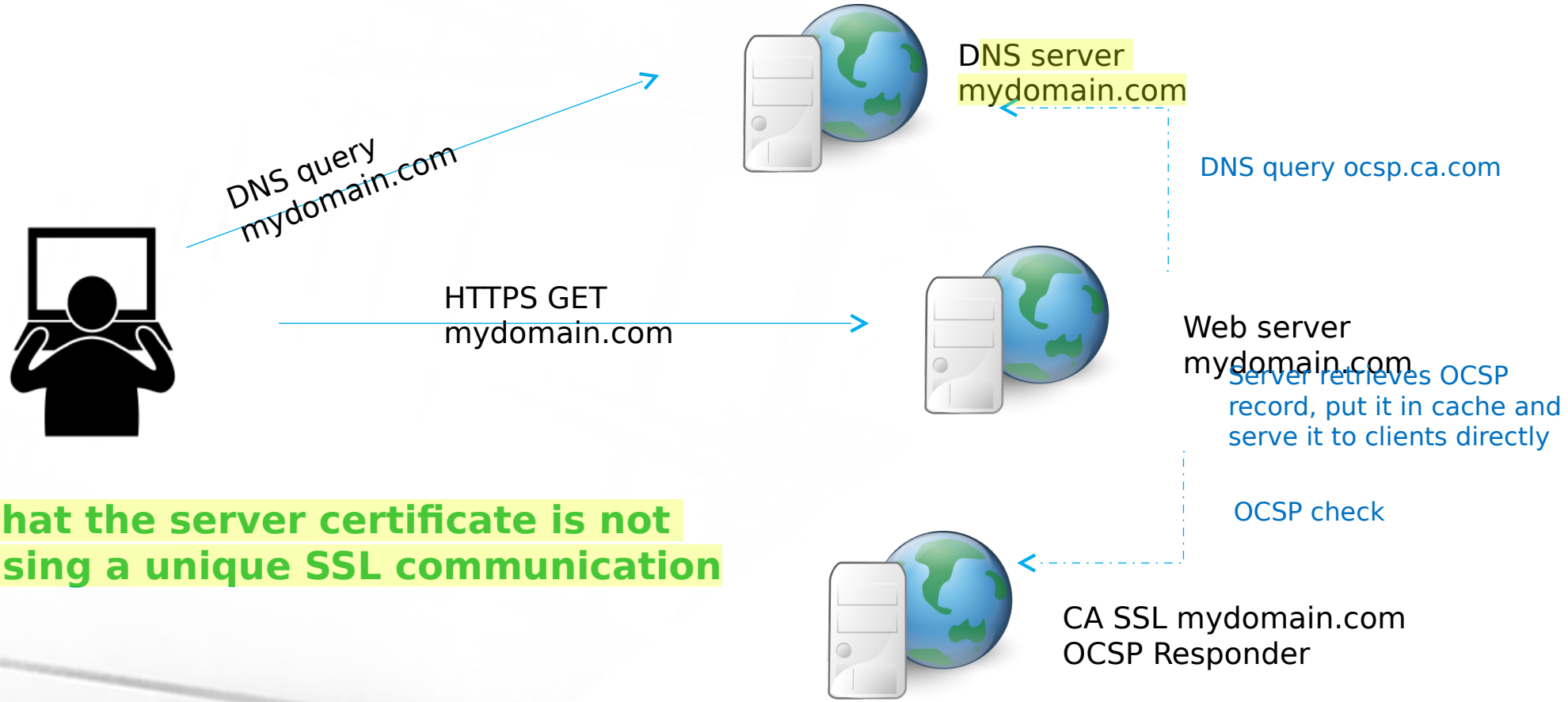
- Privacy issue: the CA can potentially track the websites you visit
- What does the browser in case of a timeout from the OCSP Responder ?
  - Stop ? Availability risk (DoS)
  - Continue ? Confidentially/integrity risk

What does Firefox (v 56.0) do today ?

By default, Firefox currently continues the connection.

OCSP Stapling : Presentation

- OCSP stapling browser validation
- « OCSP-must-staple »

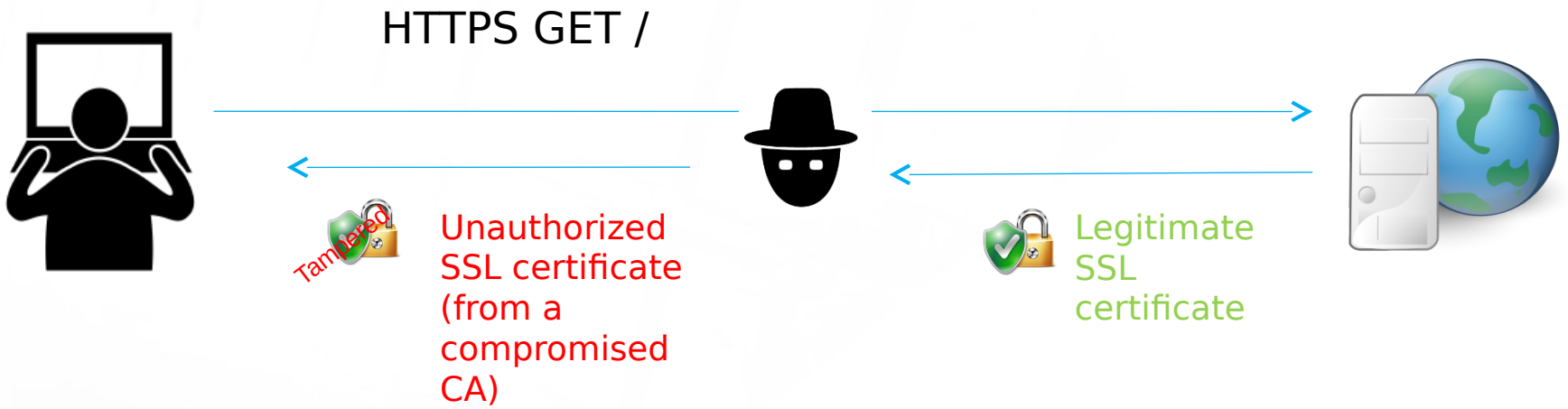


Check that the server certificate is not revoked using a unique SSL communication

HPKP : Presentation

HTTP Public Key Pinning Extension

- Without HPKP the browser will trust all the certificates signed by a CA present in the browser store when establishing a TLS connection
- With HPKP the browser will ONLY trust a list of pre-defined set of ‘pinned’ **public keys**



Fraudulent certificates – known cases

Most popular cases:

2011 - GlobalTrust.it hacked – 9 fraudulent certificates generated

2011 - DigiNotar (NL) hacked - more than 500 fraudulent certificates generated

2014 - National Informatics Centre of India – several fraudulent certificates (google) generated

2015 - CNNIC (CN) – unauthorized digital certificates for several Google domains

HPKP : Presentation

→ Mitigate MITM attack with forged certificates

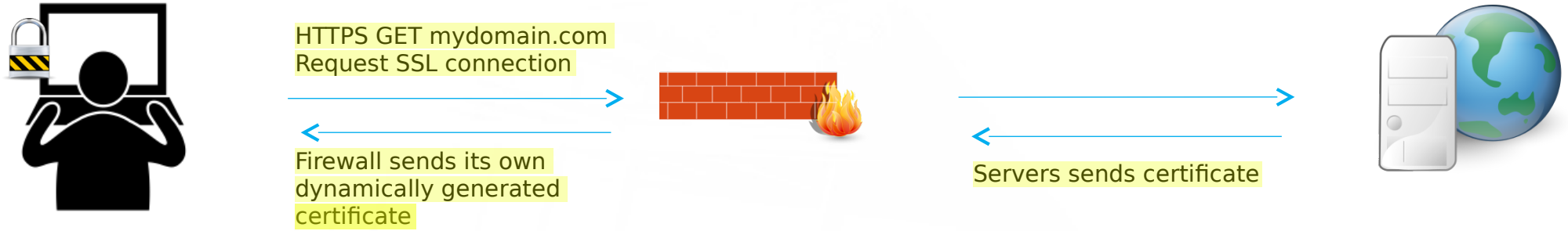
→ Detection of unauthorized certificate (from an compromised CA) AFTER the first connection

Public-Key-Pins-Report-Only: pin-sha256="base64=="; max-age=expireTime [; includeSubDomains]; report-uri="reportURI"

- At least one backup key must be pinned (in case current public key must be replaced ␣ certificate revoked)
- Alerting mechanism with the optional “report-uri” to report forbidden public keys
  - POST a “violation report” in JSON format
  - Only supported by few browsers
- Possibility to “pin” the keys of Root and intermediate CA

HPKP : Presentation

- Preloaded list exists (built-in in the browser) but no submission pages
- What about HPKP and “outbound” SSL decryption ?



- Browser should ignore the pinning in case of CA installed
- Shouldn’t conflict with « SSL decryption » (on NGFW/Forward proxy) implementation to inspect outgoing surf traffic

Firefox: security.cert_pinning.enforcement_level = 1

- 0. Pinning disabled
- 1. Allow User MITM (pinning not enforced if the trust anchor is a user inserted CA, default)
- 2. Strict. Pinning is always enforced.
- 3. Enforce test mode.

HPKP : Presentation

Limitations:

- Not supported by every browser such as Safari, IE11, Edge (under consideration),;
  - Supported by Firefox (>35), Chrome, Opera, Android
- First connection remains unprotected (TOFU)
- Hostile Pining: could be misused by a bad guy to block the access to your website (and ask ransom?)
  - The bad guy insert a HPKP header with his own public key and with a high ‘max-age’ value
  - The visitor got an error message and will not be able to visit the website until expiration of the ‘max-age’
  - Impact still occurs after the header has been corrected (persistent in the browsers)
  - Browsers decides of the maximum ‘max-age’ value – no RFC standard
- Privacy concern (super cookie)
  - Can only occur with HTTPS (not HTTP)
- Mozilla recommendation “Mandatory for maximum risk sites only - Not recommended for most site”

HTTPS protocols/ciphers suite/signature algorithms

- Protocols
  - TLS 1.3/1.2/1.1/1.0/SSLv3/SSLv2
- Ciphers Suites
- Certificates and signature algorithms (e.g. SHA256)

- Perfect Forward Secrecy (PFS)
  - Encrypted recorded communications in the past cannot be decrypted
  - Intercepted today decrypted tomorrow ?
  - Attribute of the specific key exchange mechanisms
  - Diffie-Hellman Ephemeral (DHE) or Elliptic Curves (ECDHE)

Certificate Transparency: Presentation

- Background
  - Fraudulent certificates takes time to be detected and revoked by browser vendors
- Certificate Transparency logs
- Certificate Transparency monitors
- Certificate Transparency auditors

DNS CAA

- How does it work ?
  - Use DNS entries to allow a CA to generate certificates for a domain
  - No check at the client (e.g. browser side ␣ DANE)
  - The CA/Browser Forum decided every CA must support DNS CAA checking for 09/2017
  - Not always supported by widely used DNS providers (e.g. OVH,..) – recently added into cPanel and into AWS Route 53
- Advantages
- Implementation

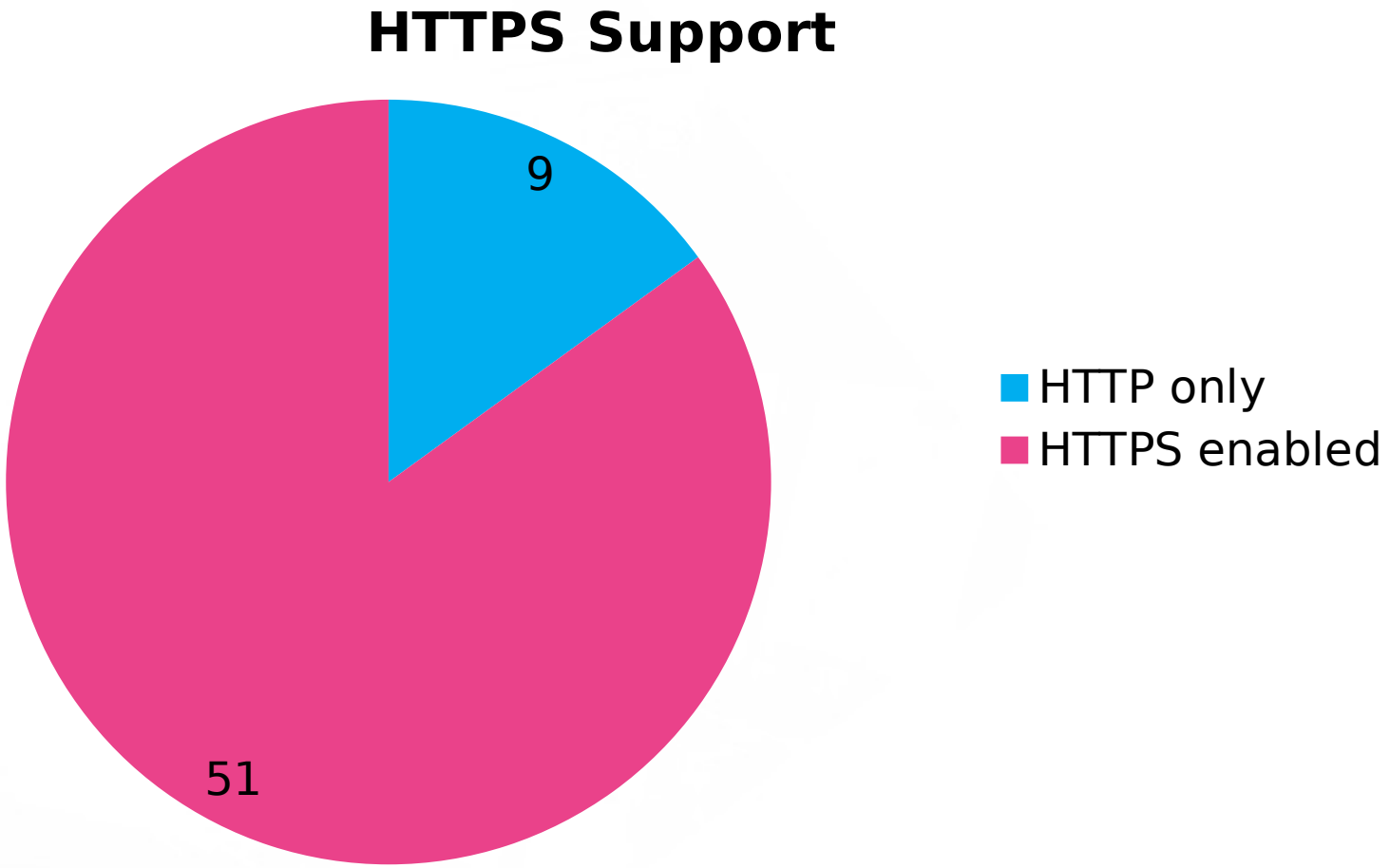
```
example.com.        CAA  0   issue       “entrust.com"
                    CAA  0   issue        "letsencrypt.org“
                    CAA  0   issuewild  "entrust.com"
                    CAA  128 iodef      "mailto:security-
incident@example.com"
beta.example.com CAA  0   issue       “digicert.com"
```

Current HTTPS implementations in Luxembourg

- Let’s see the statistics
  - Top 60 country Luxembourg TLD .lu in October 2017 (source Alexa.com)

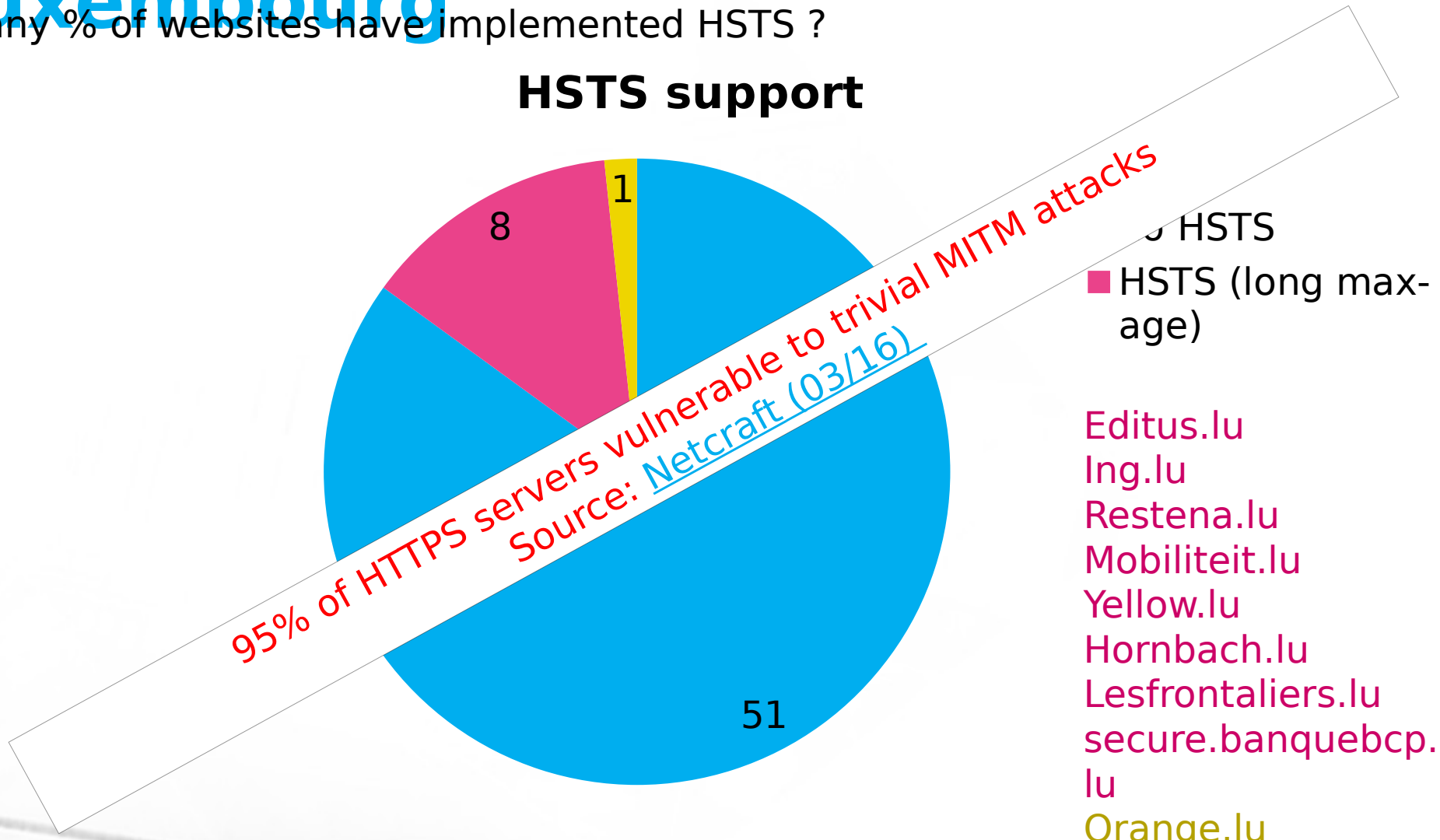
- HSTS
- HPKP
- OCSP Stapling
- DNS CAA
- Forward secrecy
- Ciphers

Current HTTPS implementations in Luxembourg



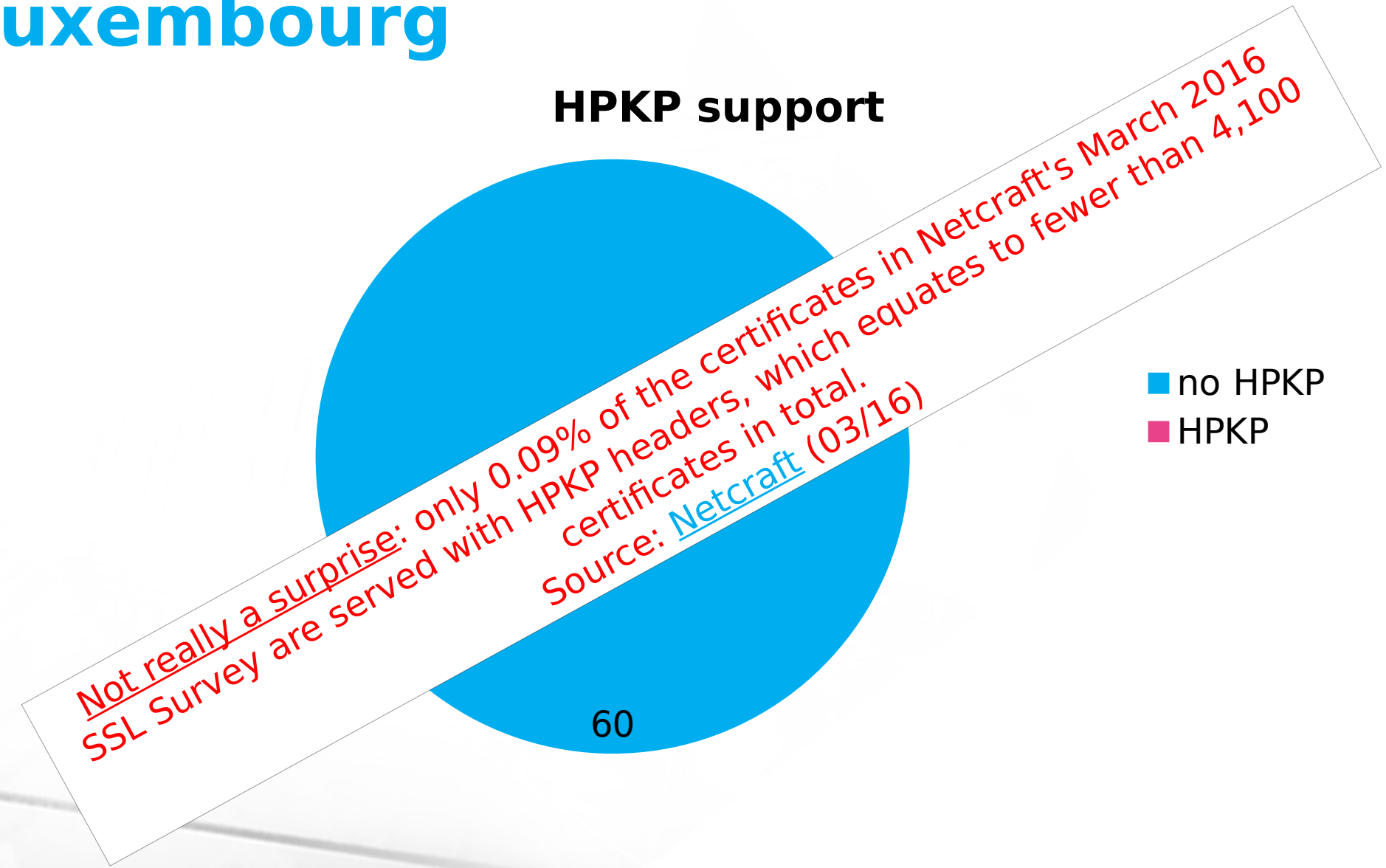
Current HTTPS implementations in Luxembourg

How many % of websites have implemented HSTS ?

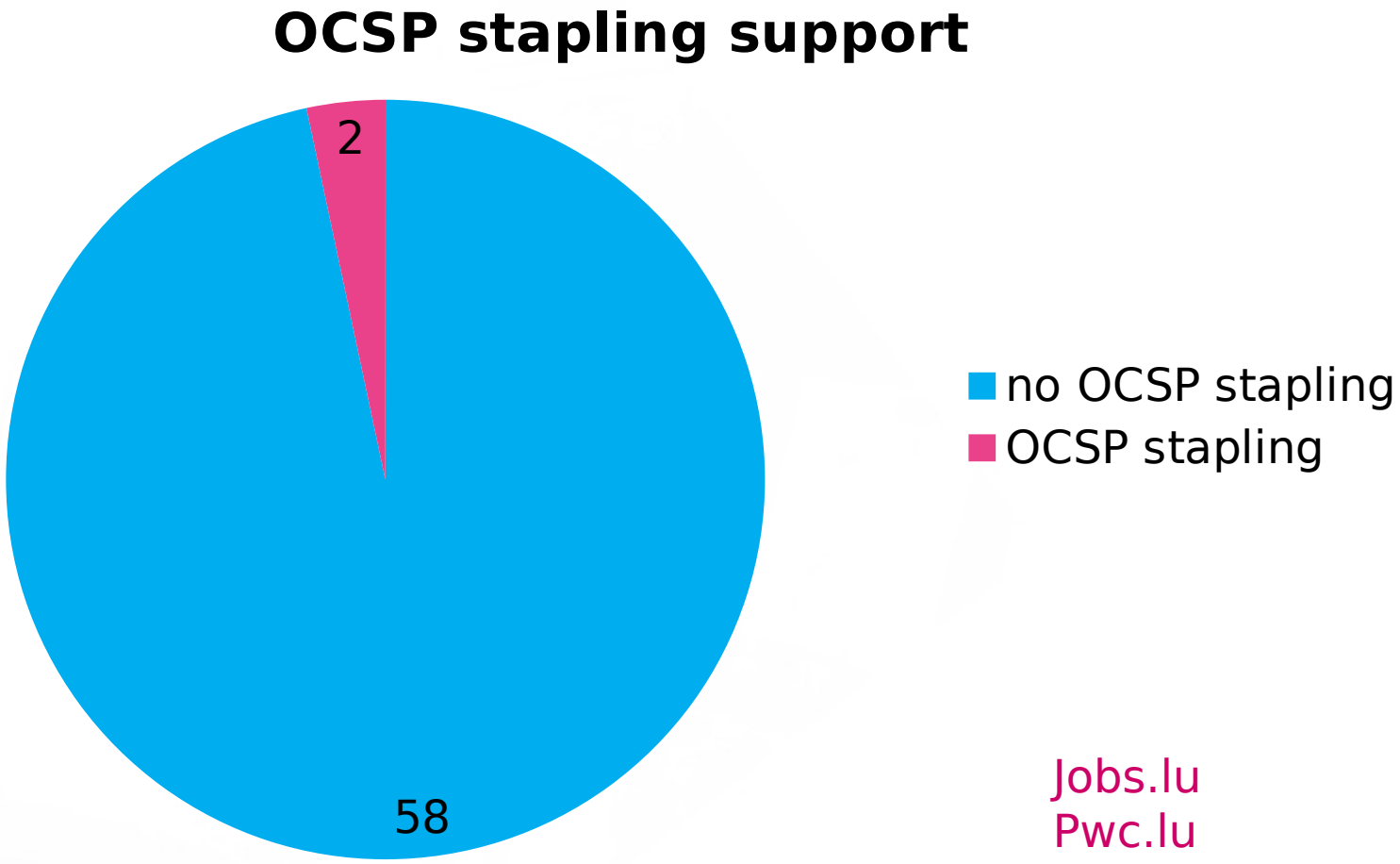


Editus.lu
Ing.lu
Restena.lu
Mobiliteit.lu
Yellow.lu
Hornbach.lu
Lesfrontaliers.lu
secure.banquebcp.lu
Orange.lu

Current HTTPS implementations in Luxembourg

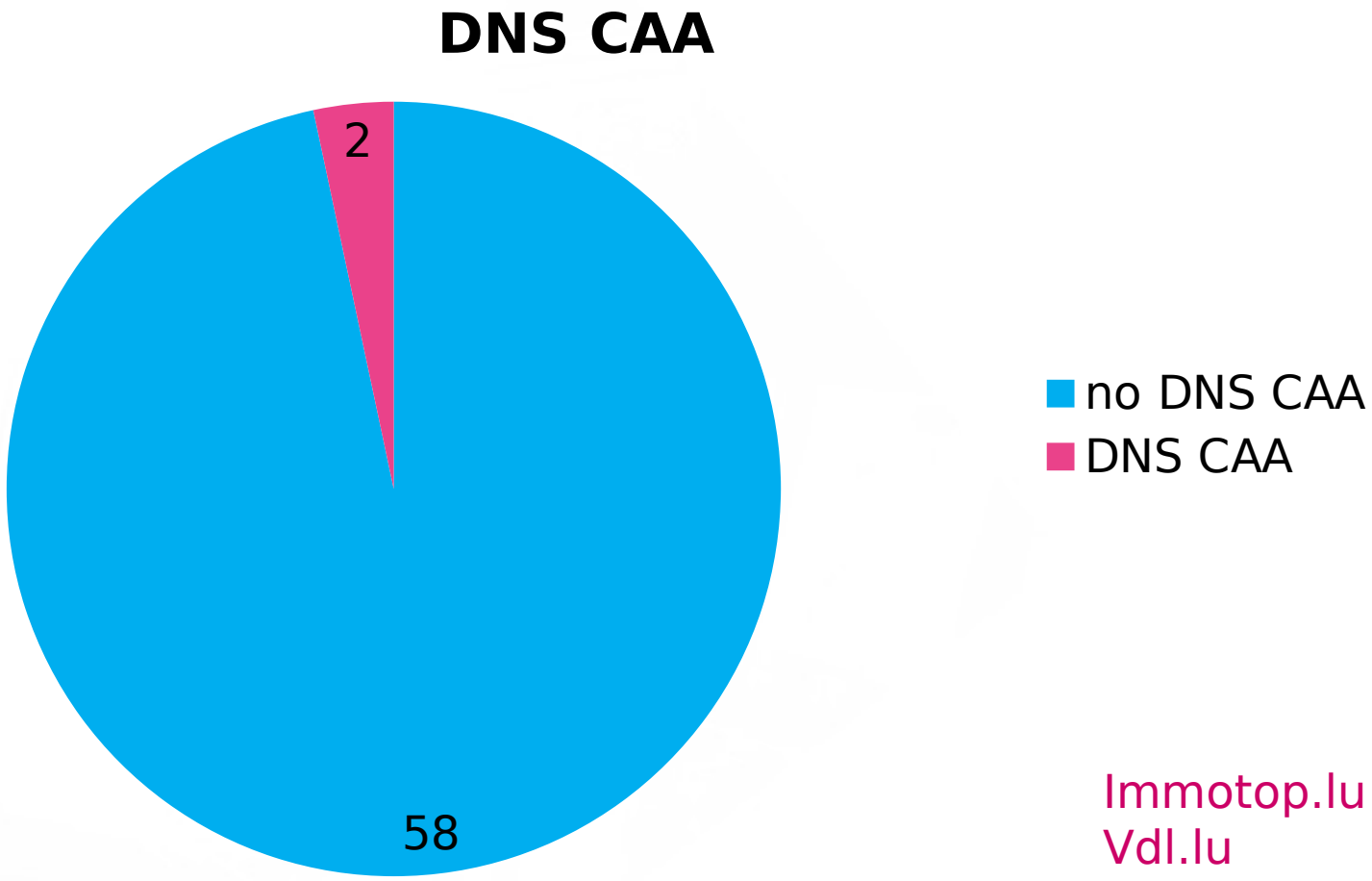


Current HTTPS implementations in Luxembourg



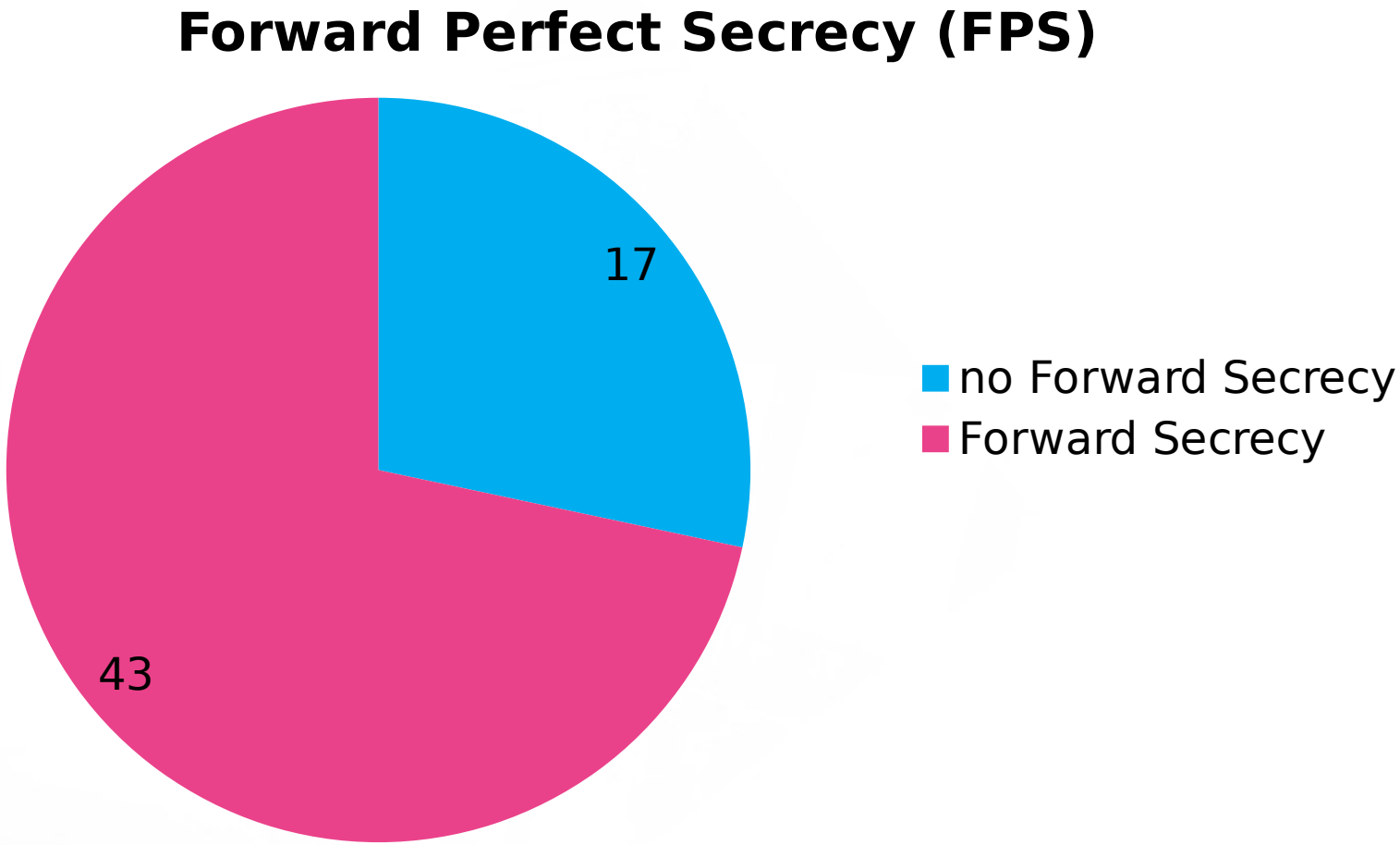
Jobs.lu
Pwc.lu

Current HTTPS implementations in Luxembourg

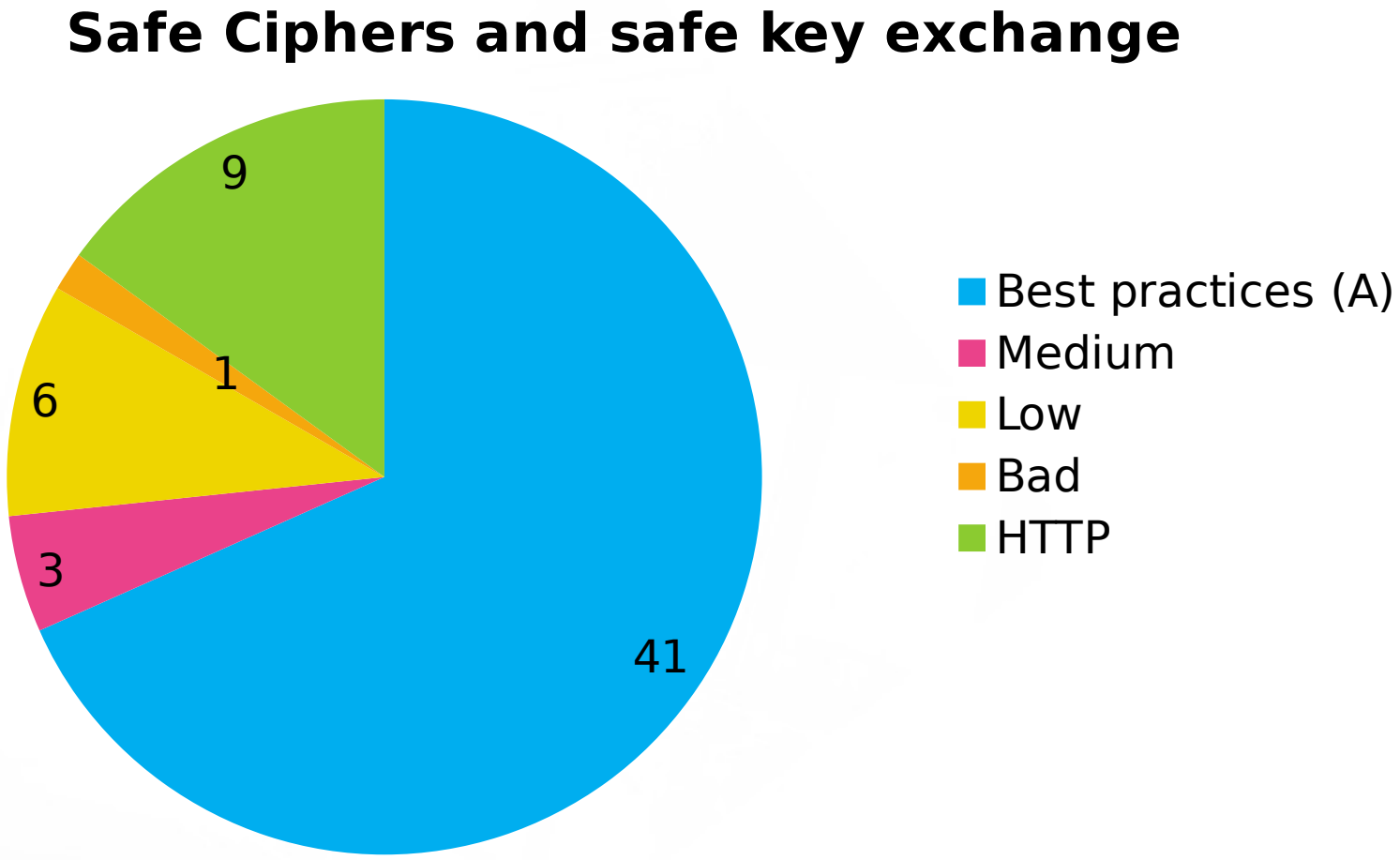


Immotop.lu
Vdl.lu

Current HTTPS implementations in Luxembourg



Current HTTPS implementations in Luxembourg




thank you
tenki
ขอขอบคุณคุณ
takk
спасибо
kam sah hamnida
dhanyavadagalu
tack
дзякуй
hvala
dziękuję
bedankt
הדות
xièxie
tanemirt
dank je
gracias
blagodaram
mèsi
rahmet
enkosi
mochchakkeram
trugarez
djere dieuf
arigatô
danke
kop khun krap
faafetai lava
manana
diolch
akun
shukriya
ありがとう
kia ora
dankon
děkuji
ačiū
grazzi
gràçie
kiitos
spas
mamnun
barka
dhanyavad
gratias ago
tau
शुक्रिया
sulpáy
tapadh leat
gràcies
dankie
chorakaloutioun
teşekkür ederim
bayarlalaa
σας ευχαριστώ
obrigada
mahalo
didi madloba
chokrane
rahmat
sagolun
murakoze
taiku
sukriya
obrigado
dakujem
misaotra
welalin
najis tuke
terima kasih
mercé
köszönöm
شكرا
asante
grazie
謝謝
mersi
sobodi
nanni
vinaka
mauruuru
nandri
merci
matondo
cảm ơn bạn
go raibh maith agat
paldies
ngiyabonga

Let’s now discuss together about it

- Webmasters
- HTTP or HTTPS website
- HSTS implementation status
  - Preload list
- Implementation issues
- Victim of target attacks
- DNS CAA implementation status
- OCSP stapling implementation status