Fortgeschrittene HTTP-Headerflags mit Fußangeln

Dirk Wetter

@drwetter

OWASP AppSecEU 15
Amsterdam, The Netherlands

OWASP
Open Web Application Security Project

whoami

- Independent Security Consultant
  - Offense / Defense / Security Project Management
  - Historical strong Unix/networking background
- OWASP Involvement
  - OWASP AppSec Research 2013
  - German OWASP Day 2012, 2014

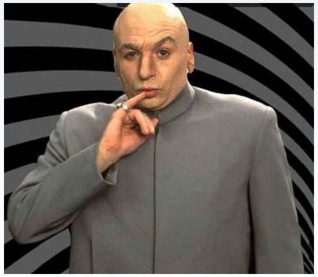
    chair
  - German Chapter Lead
  - Helping hand here/there
- Other
  - My TLS hobby: testssl.sh

```
dirks@laptop:~|1% wget -qSO /dev/null eiklaut.net
  HTTP/1.1 200 OK
  Date: Thu, 21 May 2015 22:44:42 GMT
  Content-Type: text/html
  Content-Length: 630
  Last-Modified: Fri, 29 Nov 2013 08:39:54 GMT
  Server: Apache 2.2.22 (RHEL), mod_ssl openssl 1.0.3
  Set-Cookie: PHPSESSID=44h5vc482fgiapfmit5t0n9f03; path=/;
  X-Powered-By: PHP/4.4.42
  X-UA-Compatible: IE=EmulateIE7
  Accept-Ranges: bytes
  Connection: keep-alive
dirks@laptop:~|0%
```

← Server Date

2. Where does it come from?

- Conclusion #I
  - Fingerprinting of
    - Architecture / Infrastructure
    - Patchlevel of Components
    - Maybe more
  - Evil dude: Gimme more!



- **Try this at home but not in production!**

(CC BY-NC-ND 2.0 John C. Bullas https://www.flickr.com/photos/johnbullas/532865319)

2. Where does it come from?

- Conclusion #II
  - Client side perspective:
    - What else can be done to improve security?

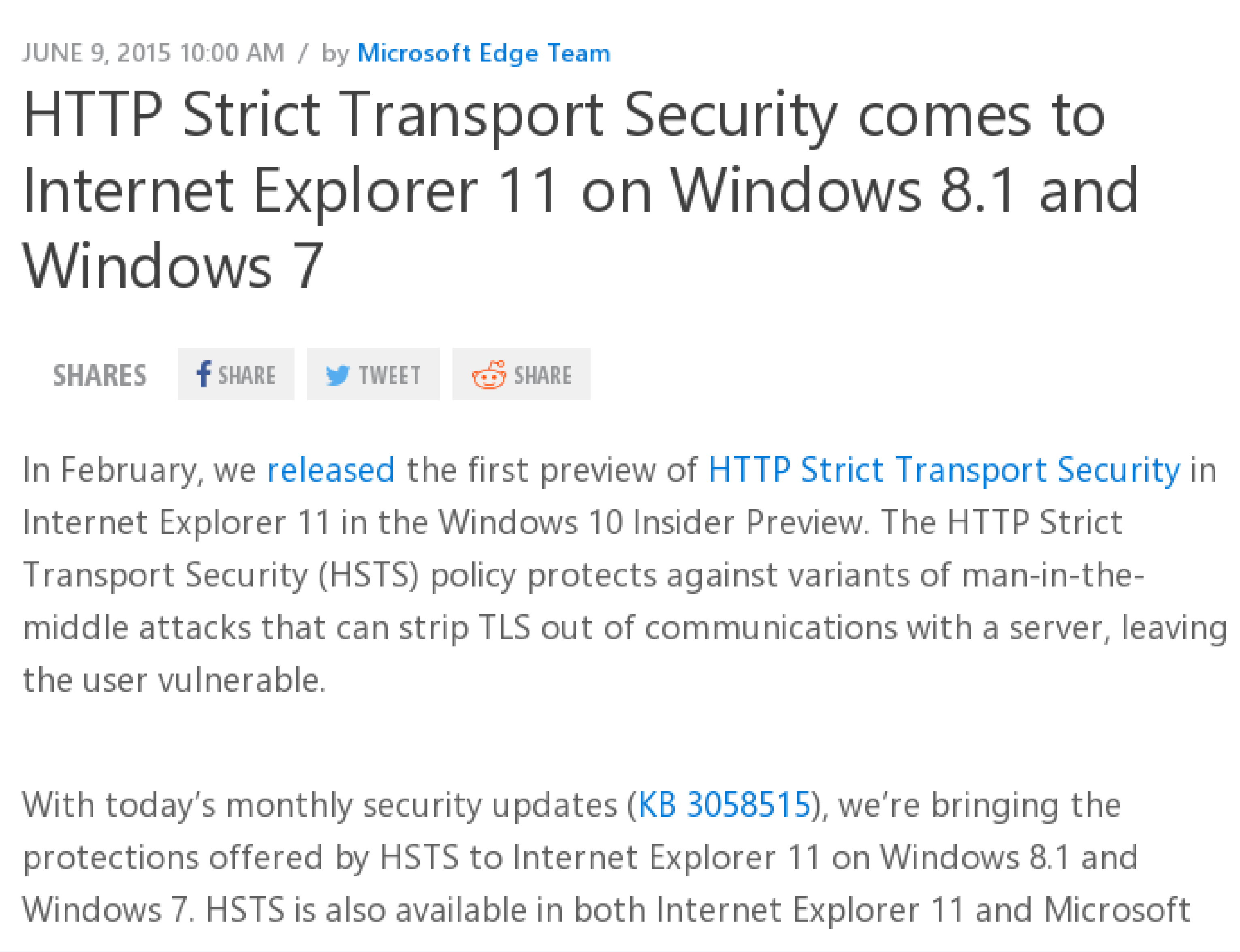
→ Browser Perspective

3. Theory

- Transport encryption: HSTS (RFC 6797)
  - HTTP `Strict-Transport-Security`
  - @Browser : Within `max-age` never ever use plain http, maybe `includeSubDomains`
  - `max-age:` [seconds]
    - Recommended > ½ year: 3600 • 24 • 181
- Browser support ok *)
  - ✔ Chrome >=4
  - ✔ Firefox >= 4
  - ✔ Safari >=7, Opera >=12.1
  - ~~IE 11 on W10 only!~~

*) see http://caniuse.com/#feat=stricttransportsecurity

3. Theory



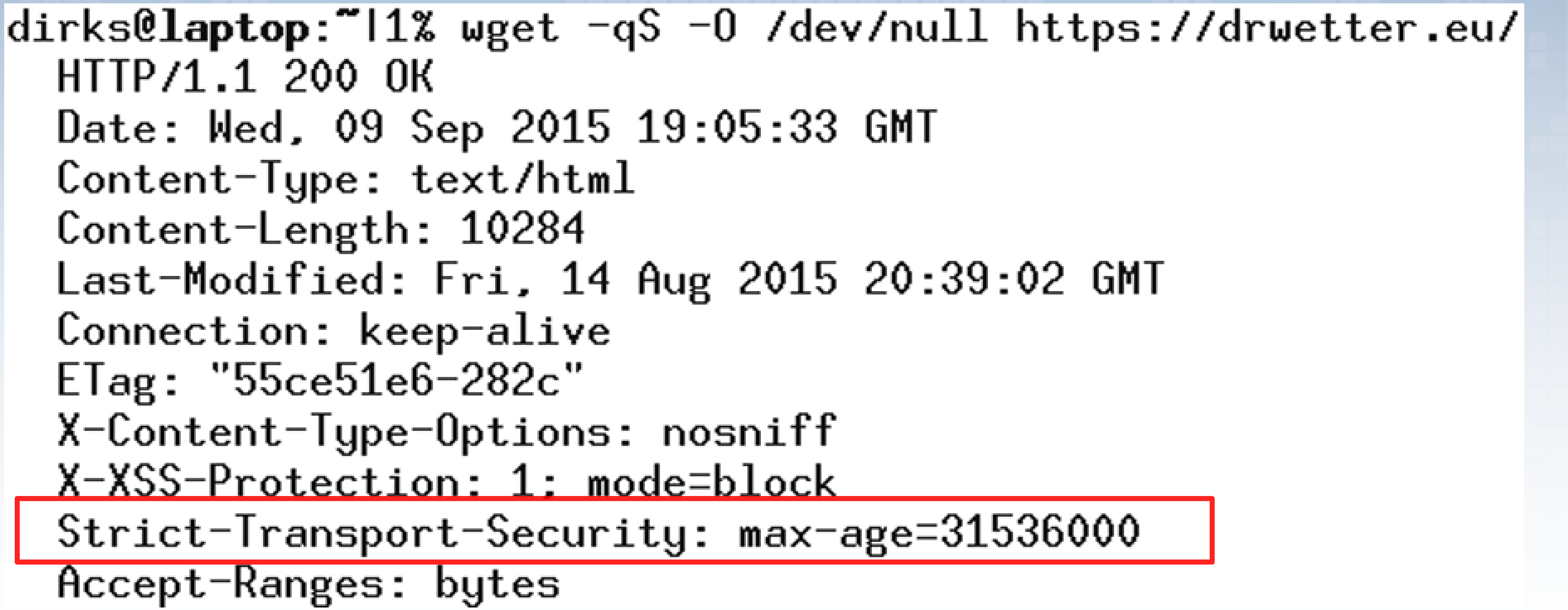
JUNE 9, 2015 10:00 AM / by Microsoft Edge Team

HTTP Strict Transport Security comes to Internet Explorer 11 on Windows 8.1 and Windows 7

In February, we released the first preview of HTTP Strict Transport Security in Internet Explorer 11 in the Windows 10 Insider Preview. The HTTP Strict Transport Security (HSTS) policy protects against variants of man-in-the-middle attacks that can strip TLS out of communications with a server, leaving the user vulnerable.

With today's monthly security updates (KB 3058515), we're bringing the protections offered by HSTS to Internet Explorer 11 on Windows 8.1 and Windows 7. HSTS is also available in both Internet Explorer 11 and Microsoft

3. Theory



```
dirks@laptop:~|1% wget -qS -O /dev/null https://drwetter.eu/
  HTTP/1.1 200 OK
  Date: Wed, 09 Sep 2015 19:05:33 GMT
  Content-Type: text/html
  Content-Length: 10284
  Last-Modified: Fri, 14 Aug 2015 20:39:02 GMT
  Connection: keep-alive
  ETag: "55ce51e6-282c"
  X-Content-Type-Options: nosniff
  X-XSS-Protection: 1; mode=block
  Strict-Transport-Security: max-age=31536000
  Accept-Ranges: bytes
```

3. Theory

- Transport encryption: HSTS
  - One problem though

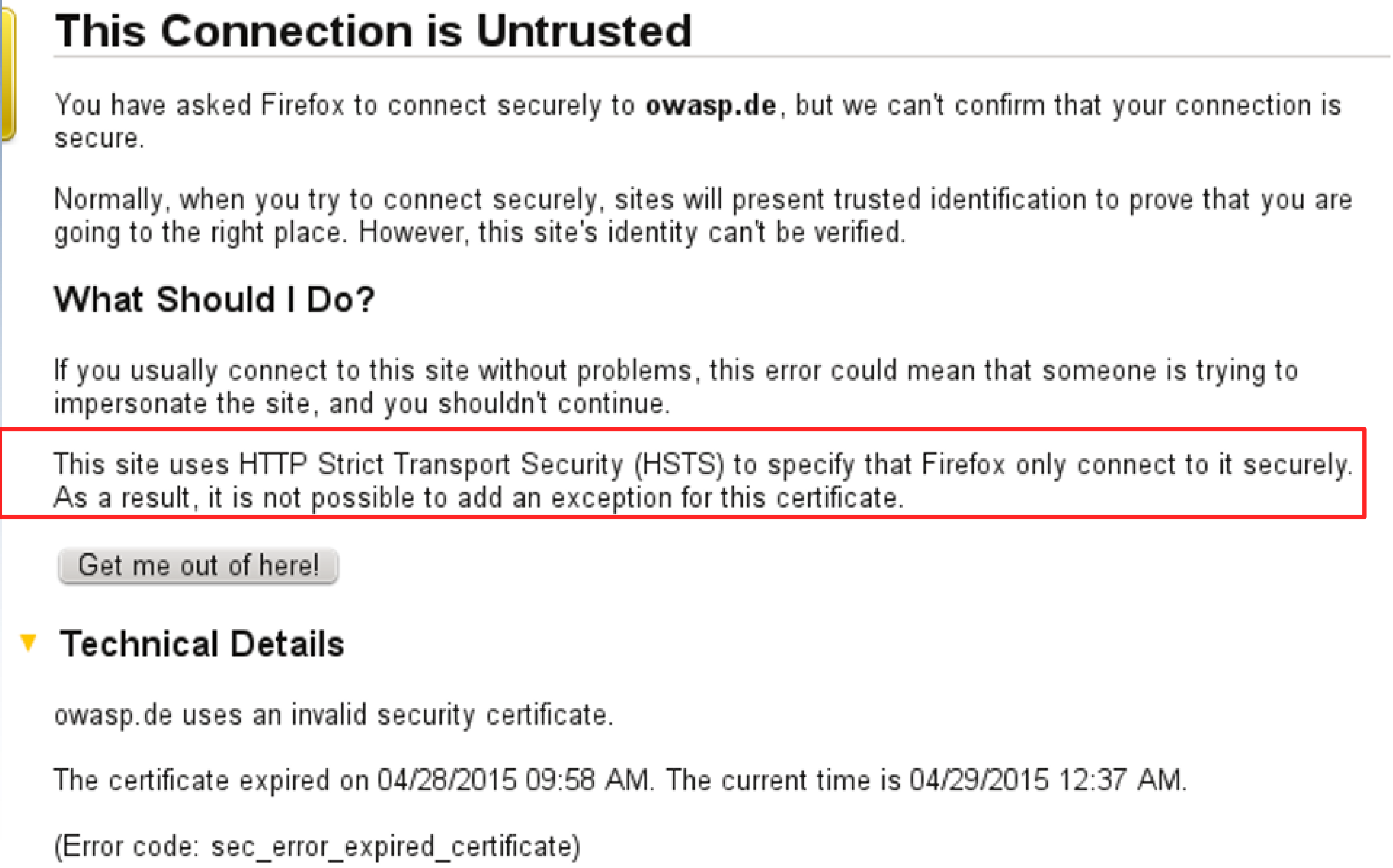
TOFU!

  - `preload:` http://hstspreload.appspot.com/

DryPot / GDFL 2.0/ CC BY-SA 3.0 http://en.wikipedia.org/wiki/Tofu#/media/File:Japanese_SilkyTofu_(Kinugoshi_Tofu).JPG

3. Theory

- Transport encryption: HSTS
  - Pitfalls
    - Change of mind (http!) within time specified
    - Careful with `includeSubDomains` for TLD!
    - Certificate expired (example Firefox)



This Connection is Untrusted

You have asked Firefox to connect securely to **owasp.de**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

This site uses HTTP Strict Transport Security (HSTS) to specify that Firefox only connect to it securely. As a result, it is not possible to add an exception for this certificate.

Technical Details

owasp.de uses an invalid security certificate.

The certificate expired on 04/28/2015 09:58 AM. The current time is 04/29/2015 12:37 AM.

(Error code: sec_error_expired_certificate)

3. Theory

- Transport encryption: HSTS
  - Solution to Pitfalls
    - set `max-age` to 0 (see RFC 6797, 6.1.1)
      - worked w/ FF
      - Chrome?
    - order new certificate ;-)

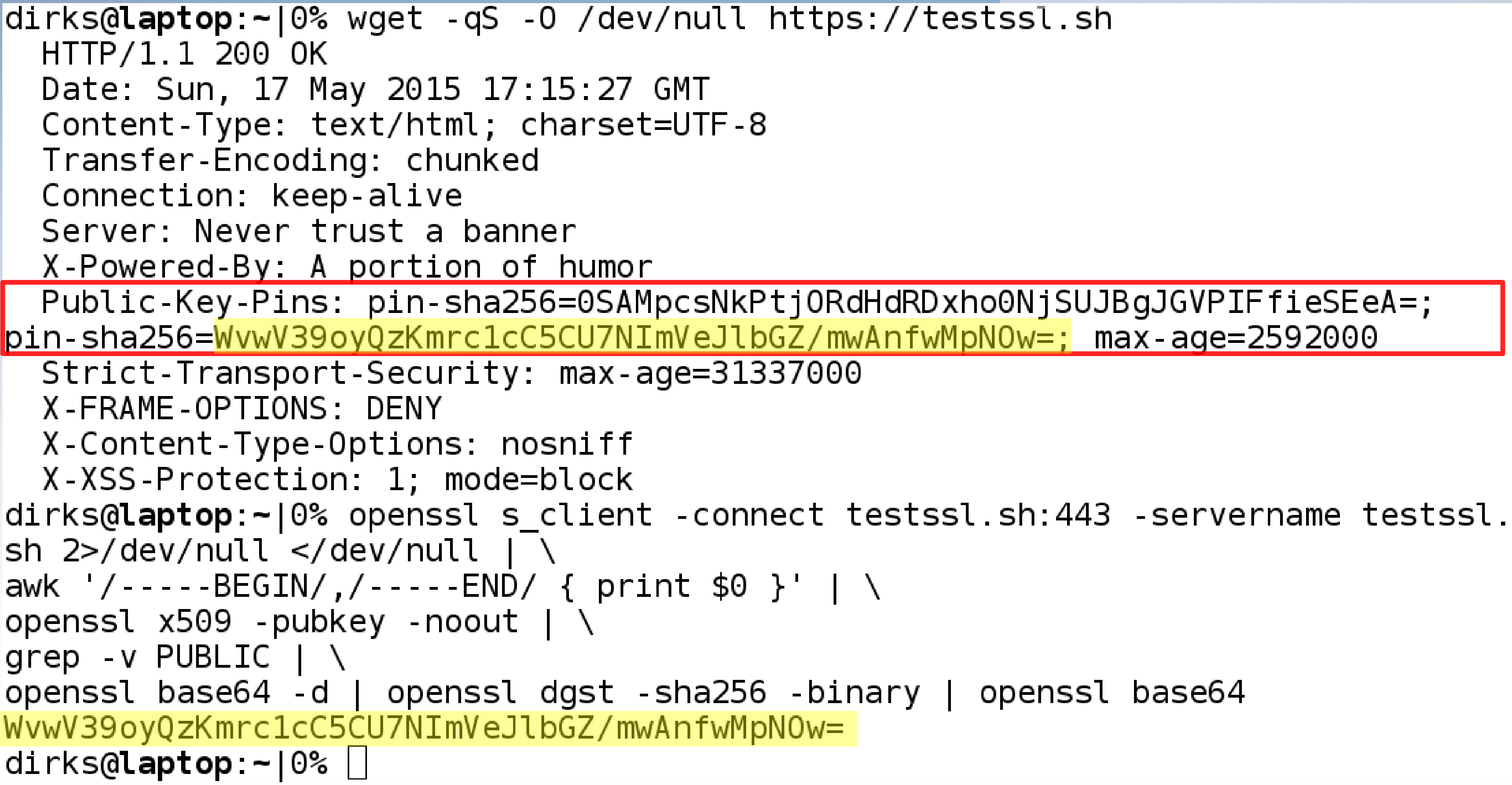
3. Theory

- IETF ~~Draft~~ RFC 7469

  Public-Key-Pins / Public-Key-Pins-Report-Only

  - Certificate pinning a.k.a. HPKP
  - @Browser: Within max-age remember keys and do not except anything else
  - pin-sha256=<base64str> …
  - report-uri=<json-POST-URI> !
  - IncludeSubDomains
  - Violations (json POST URI):
    - Public-Key-Pins report-uri=<URI>
    - Public-Key-Pins-Report-Only report-uri=<URI>

3. Theory



```
dirks@laptop:~|0% wget -qS -O /dev/null https://testssl.sh
  HTTP/1.1 200 OK
  Date: Sun, 17 May 2015 17:15:27 GMT
  Content-Type: text/html; charset=UTF-8
  Transfer-Encoding: chunked
  Connection: keep-alive
  Server: Never trust a banner
  X-Powered-By: A portion of humor
  Public-Key-Pins: pin-sha256=0SAMpcsNkPtjORdHdRDxho0NjSUJBgJGVPIFfieSEeA=;
pin-sha256=WvwV39oyQzKmrc1cC5CU7NImVeJlbGZ/mwAnfwMpNOw=; max-age=2592000
  Strict-Transport-Security: max-age=31337000
  X-FRAME-OPTIONS: DENY
  X-Content-Type-Options: nosniff
  X-XSS-Protection: 1; mode=block
dirks@laptop:~|0% openssl s_client -connect testssl.sh:443 -servername testssl.
sh 2>/dev/null </dev/null | \
awk '/-----BEGIN/,/-----END/ { print $0 }' | \
openssl x509 -pubkey -noout | \
grep -v PUBLIC | \
openssl base64 -d | openssl dgst -sha256 -binary | openssl base64
WvwV39oyQzKmrc1cC5CU7NImVeJlbGZ/mwAnfwMpNOw=
dirks@laptop:~|0% 
```

3. Theory

- Goodbye (un)lawful / whatsoever interception ;-)
  - RFC: **UAs MUST close the connection to a host upon Pin Failure**

- Ok, some constraints...
  - (quality of encryption)
  - Local CAs
  - TOFU

Picture courtesy of Submessenger @ www.bullshido.net

3. Theory

- HPKP: which keys?
  - Best: pins of >= two keys
    - actual
    - Pin of backup key in pocket / safe
    - Careful with issuer pinning
  - Locking out?
    - `max-age=0` ?? Draft / RFC

3. Theory

- HPKP, browser support:
  - ✔ Chrome 35+
  - ✔ Firefox 35+
  - ✔ Opera 31+
  - Safari
  - IE 1x / Edge

3. Theory

Was: per site

Now: per page

3. Theory

- `X-Content-Type-Options: nosniff`
  - Originally designed for IE < 8
    - MIME sniffing! Ignored Content-Type completely
      - Similar unix `file` (or MIME libs → file uploads)
    - User controlled content
      - chameleon files, picture XSS
      - HTML/JS in PS, PNG etc.

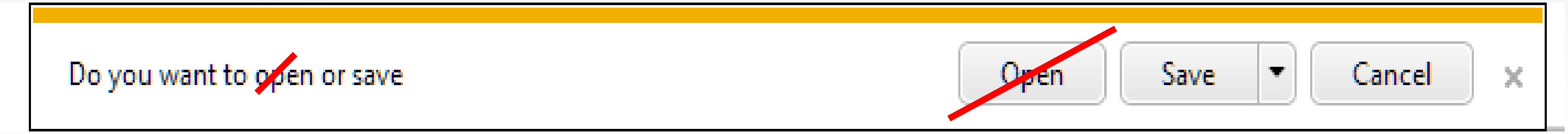
    - Attention:
      - Compatibility view of IE
        `X-UA-Compatible: IE=EmulateIE7`

Warning for Aliens, liftarn,openclipart.org/detail/2881/warning-for-aliens

T-800: by Noble0, CC BY SA 3.0 / r.wikipedia.org/wiki/Fichier:Old_Artificial_Profile_Avatar.jpg

3. Theory

- Safer dl'ing:
  - `Content-Disposition: attachment`
  - `X-Download-Options : noopen`



Do you want to ~~open~~ or save

CC-BY_SA 2.0 Magnus Manske https://commons.wikimedia.org/wiki/File:Condom_gun.jpg

3. Theory

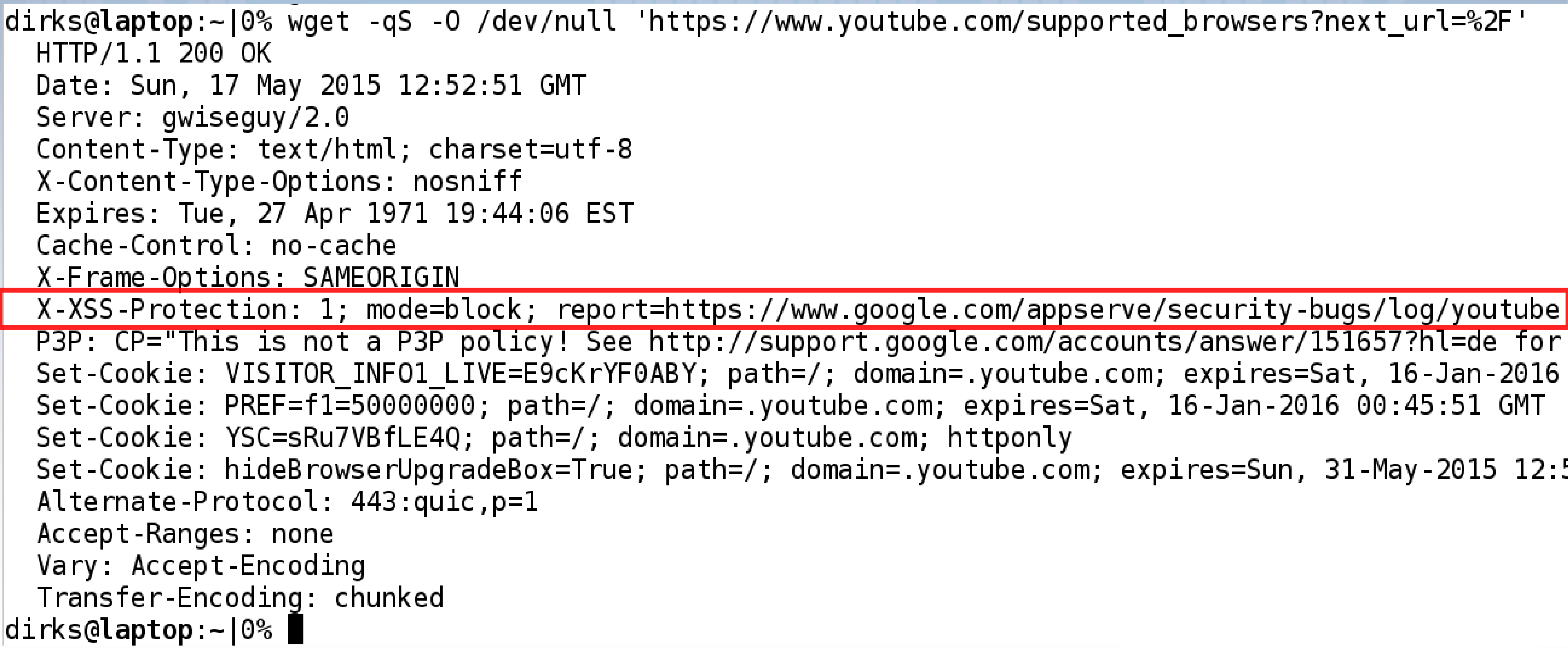
- `X-FRAME-OPTIONS`
  - `DENY | SAMEORIGIN | ALLOW-FROM` *`uri`*
  - Clickjacking 'n friends
  - Against framing **your** page

  - Attention:
    - Business reasons for framing
    - Application (e.g. Typo3 backend)
    - Chrome/Webkit? don't give a damn: `ALLOW-FROM`

3. Theory

- XSS
  - `X-XSS-Protection`
    - `0` → off (https://www.facebook.com/)
    - `1` → on, browser engine can sanitize
    - `1; mode=block` → better: no rendering
    - Good thing (again):
      - `report=<JSON POST URI>`
  - Support:
    - no Firefox
    - ✔ Chrome/Webkit
    - ✔ IE

3. Theory



```
dirks@laptop:~|0% wget -qS -O /dev/null 'https://www.youtube.com/supported_browsers?next_url=%2F'
  HTTP/1.1 200 OK
  Date: Sun, 17 May 2015 12:52:51 GMT
  Server: gwiseguy/2.0
  Content-Type: text/html; charset=utf-8
  X-Content-Type-Options: nosniff
  Expires: Tue, 27 Apr 1971 19:44:06 EST
  Cache-Control: no-cache
  X-Frame-Options: SAMEORIGIN
  X-XSS-Protection: 1; mode=block; report=https://www.google.com/appserve/security-bugs/log/youtube
  P3P: CP="This is not a P3P policy! See http://support.google.com/accounts/answer/151657?hl=de for
  Set-Cookie: VISITOR_INFO1_LIVE=E9cKrYF0ABY; path=/; domain=.youtube.com; expires=Sat, 16-Jan-2016
  Set-Cookie: PREF=f1=50000000; path=/; domain=.youtube.com; expires=Sat, 16-Jan-2016 00:45:51 GMT
  Set-Cookie: YSC=sRu7VBfLE4Q; path=/; domain=.youtube.com; httponly
  Set-Cookie: hideBrowserUpgradeBox=True; path=/; domain=.youtube.com; expires=Sun, 31-May-2015 12:5
  Alternate-Protocol: 443:quic,p=1
  Accept-Ranges: none
  Vary: Accept-Encoding
  Transfer-Encoding: chunked
dirks@laptop:~|0%
```

3. Theory

OLD

`X-Content-Security-Policy` (FF < 23)
`X-WebKit-CSP` (Chrome < 25)

- XSS, again
  - `Content-Security-Policy` (+ related) (`Content-Security-Policy-Report-Only`)
    - not trivial!
      - Cooperation of development, JS frameworks, templates, trackers, objects embedded, ...
    - ~two parts
      1. **Policy directive** (*-src)
      2. **source value(s)**
    - Concat as much pairs as you want
      - Separation by semicolon

3. Theory

- XSS, again
  - Content-Security-Policy
    - Policy directive

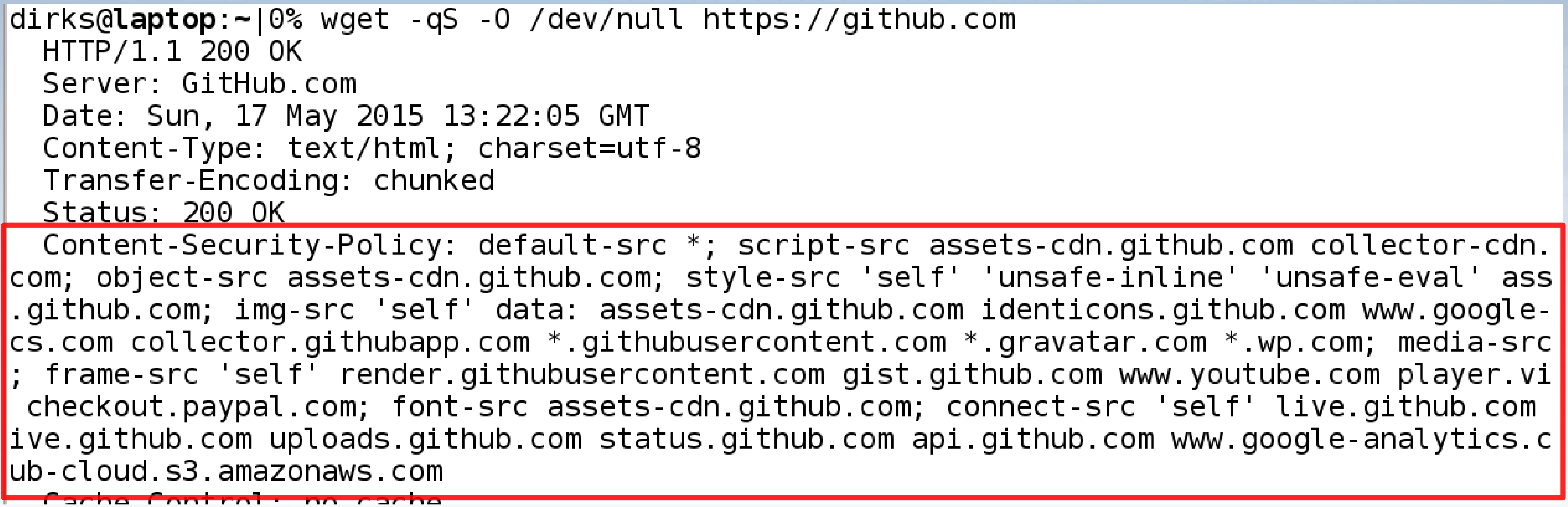
      @Browser: which content are you allowed to render?
      - » script-src
      - » img-src
      - » style-src
      - » frame-src (father option to X-FRAME-OPTIONS)
      - » font-src
      - » media-src (video, audio)
      - » default-src
      - » ...

3. Theory

- XSS, again
  - Content-Security-Policy
    - Source value: @Browser: this are you allowed to do with directive
      - 'none'
      - 'self'
      - 'unsafe-inline' → Attention: standard defense XSS
      - 'unsafe-eval' → bad
      - *Uri*
        - » Careful w/ * → (all hosts)!!!
  - Violations (json POST URI):
    - Content-Security-Policy <key value> *report-uri <URI>*
  - Support
    - Firefox 33+
    - Chrome 40+
    - IE / Edge: only fragmentary
  - Shameless plug: OWASP TT 2013

See https://www.owasp.org/index.php/Content_Security_Policy_Cheat_Sheet

3. Theory



```
dirks@laptop:~|0% wget -qS -O /dev/null https://github.com
  HTTP/1.1 200 OK
  Server: GitHub.com
  Date: Sun, 17 May 2015 13:22:05 GMT
  Content-Type: text/html; charset=utf-8
  Transfer-Encoding: chunked
  Status: 200 OK
  Content-Security-Policy: default-src *; script-src assets-cdn.github.com collector-cdn.
com; object-src assets-cdn.github.com; style-src 'self' 'unsafe-inline' 'unsafe-eval' ass
.github.com; img-src 'self' data: assets-cdn.github.com identicons.github.com www.google-
cs.com collector.githubapp.com *.githubusercontent.com *.gravatar.com *.wp.com; media-src
; frame-src 'self' render.githubusercontent.com gist.github.com www.youtube.com player.vi
 checkout.paypal.com; font-src assets-cdn.github.com; connect-src 'self' live.github.com
ive.github.com uploads.github.com status.github.com api.github.com www.google-analytics.c
ub-cloud.s3.amazonaws.com
  Cache-Control: no-cache
```

3. Theory

- CSP v2 (~~still W3C draft~~)

2+1=2

1. Whitelist scripts: Hash support
   - Inline Code:
     ```
     <script>
          notevilstuff
     </script>
     ```
   - Header: `Content-Security-Policy: script-src 'sha256-<base64 encoded hash(“notevilstuff“))>';`

2. Whitelist script: Nonce
   - Inline Code:
     ```
     <script nonce='n0n53'>
          notevilstuff
     </script>
     ```
   - Header: `Content-Security-Policy: script-src 'nonce-n0n53';`

3. Theory

- CSP v2 ~~(still W3C draft)~~
  - 3./4. Same whitelists with `style-src`
  - 5. `frame-ancestors`

- Support:
  - ✔Firefox 33+
  - ✔Chrome 40+
  - IE / Edge

4. Where to set / unset?

- Depends...
  - Your access
  - Your hat on your head

- In principle 4 possibilities
  - application
  - application server configuration

    `{server,web}.xml`, `web.config`, `php.ini` etc.
  - web server (configuration)
  - reverse proxy / load balancer / WAF

4. Where to set / unset?

- What's the best place?
  - KISS
    - What ever is the easiest for you
    - As long as the application still works
  - Best
    - At the root of the cause

5. Conclusion

- **Add security headers!!**
  - There are low hanging fruits!

- **Disclaimer: Obscurity & Security**
  - Protection against shodan search
  - But: Criminals throw everything @ target
  - Do PROPER defense!
    - **Proper secure defaults!**
    - Remove chatty lines && PATCH!

5. Conclusion

- Test it!
  - Thorough functional tests!
    - **BROWSERS!!**
  - Look for complaints (CSP+HPKP)
    - Browser Console
    - Report-uri
      - https://report-uri.io/
    - Basically: Attack surface!

@drwetter