MURAENA

antisnatchor & ohpe present

THE UNEXPECTED PHISH

OUTLINE

- Demystifying 2FA
- Reverse Proxy strikes back
  - Reversing anti-reverse proxy
- The art of instrumenting stolen web sessions
  - Bypassing anti-browser instrumentation
  - Automating post-phishing activities
- Full-chain demos on GSuite, GitHub, Dropbox
- Upcoming challenges

DEMYSTIFYING 2FA: NON-U2F MYTHS

- You would immediately realise something dodgy is going on if you receive a 2FA challenge/token for an action you did not perform
- If attackers know your credentials, they can't login
- If you set up 2FA you are more secure (even via SMS)

DEMYSTIFYING 2FA: NON-U2F MYTHS

- You would immediately realise something dodgy is going on if you receive a 2FA challenge/token for an action you did not perform

- Potentially true, but it doesn't consider scenarios where both credentials and tokens are phished in real-time

- If attackers know your credentials, they can't login

- It minimises only password spraying and other dictionary related attacks (Captcha would be enough)

DEMYSTIFYING 2FA

- *If you set up 2FA you are more secure (even via SMS)*

DEMYSTIFYING 2FA: NON-U2F MYTHS

- Universal Two Factor (U2F) is the only 2FA solution that offers protection from phishing
  - Crypto challenge with the web origin
- Bypassed in early 2018 via WebUSB by @marver and @antisnatchor
- OffensiveCon talk at: youtube.com/watch?v=pUa6nWWTO4o

REVERSE PROXY TO THE RESCUE

- Most 2FA solutions rely on token submission via a web form: SMS, Push/OTP based, Google/CompanyX authenticators
- After credentials are verified, the server triggers the 2FA action and waits for user token submission
- A smart reverse proxy can then be used to:
  - intercept all the traffic
  - fulfil requests that trigger the 2FA action
  - pass post-2FA login session cookies to an instrumented browser that hijacks the victim's session

REVERSE PROXY TO THE RESCUE

- There is no magic here (and don’t say you didn’t know):

REVERSE PROXY TO THE RESCUE

- Reverse proxies exist from over 20 years
- Technical Trends in Phishing Attacks (US CERT) - 2005

https://www.us-cert.gov/sites/default/files/publications/phishing_trends0511.pdf

3.4.4 Man-in-the-Middle Attacks

Man-in-the-middle attacks define a broad class of potential attacks in which an attacker is able to intercept, read, and modify communications between two other parties without their knowledge. As related to phishing, a man-in-the-middle attack involves an attacker serving as a proxy between a user and an online commerce site. The attacker potentially has access to all authentication and account information, including an opportunity to hijack credentials used in two-factor authentication.

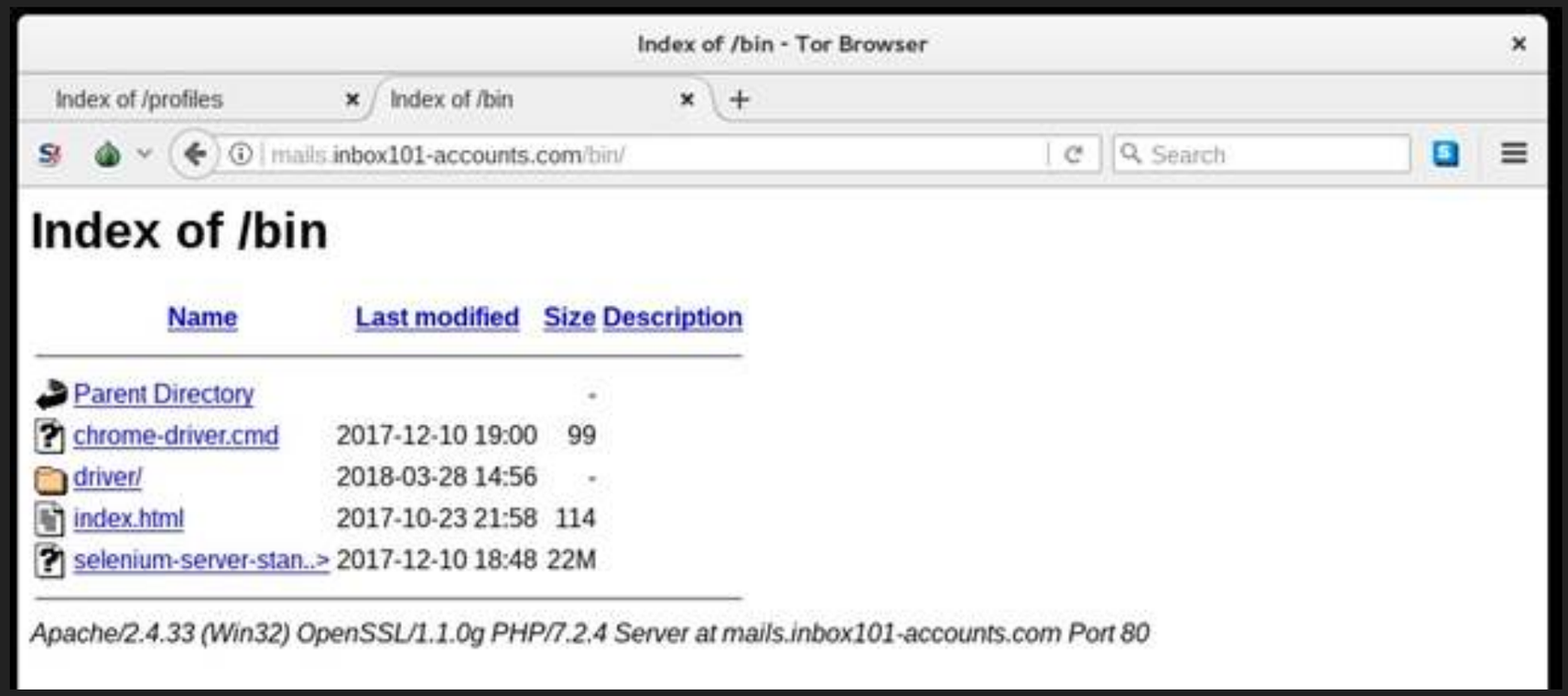
- So how did the situation evolved from 2005?

REVERSE PROXY TO THE RESCUE

- In the last 4/5 years they have been hot again and used for offensive purposes:
  - (2015) https://labs.portcullis.co.uk/blog/blood-in-the-water-phishing-with-beef/
  - (2017) http://www.chokepoint.net/2017/03/reverse-proxy-phishing-with-valid.html
  - (2017) https://breakdev.org/evilginx-advanced-phishing-with-two-factor-authentication-bypass/

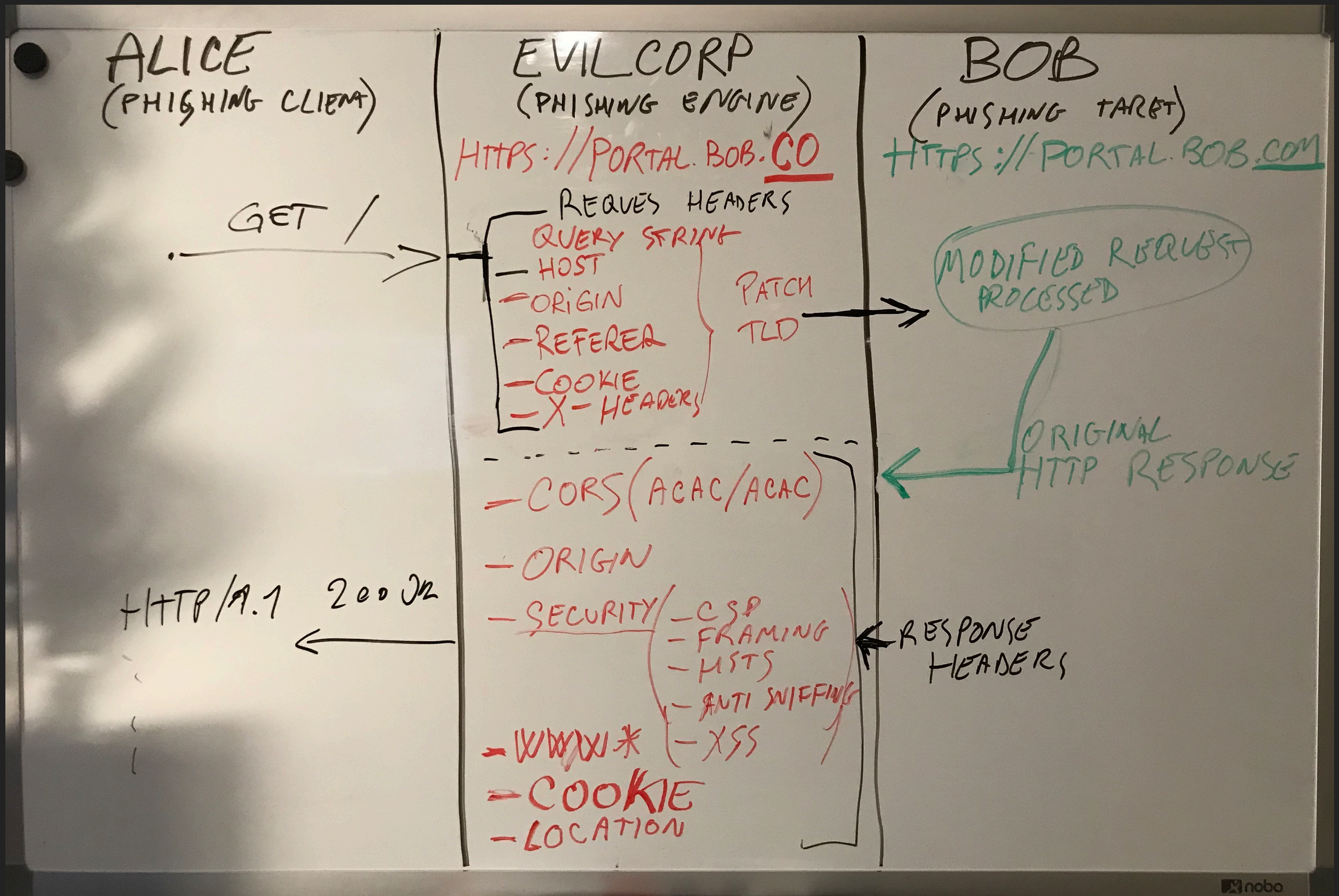
REVERSE PROXY TO THE RESCUE

- And as expected on APTs:
  - (2017) https://citizenlab.ca/2017/02/nilephish-report/
  - (2018) https://www.amnesty.org/en/latest/research/2018/12/when-best-practice-is-not-good-enough/

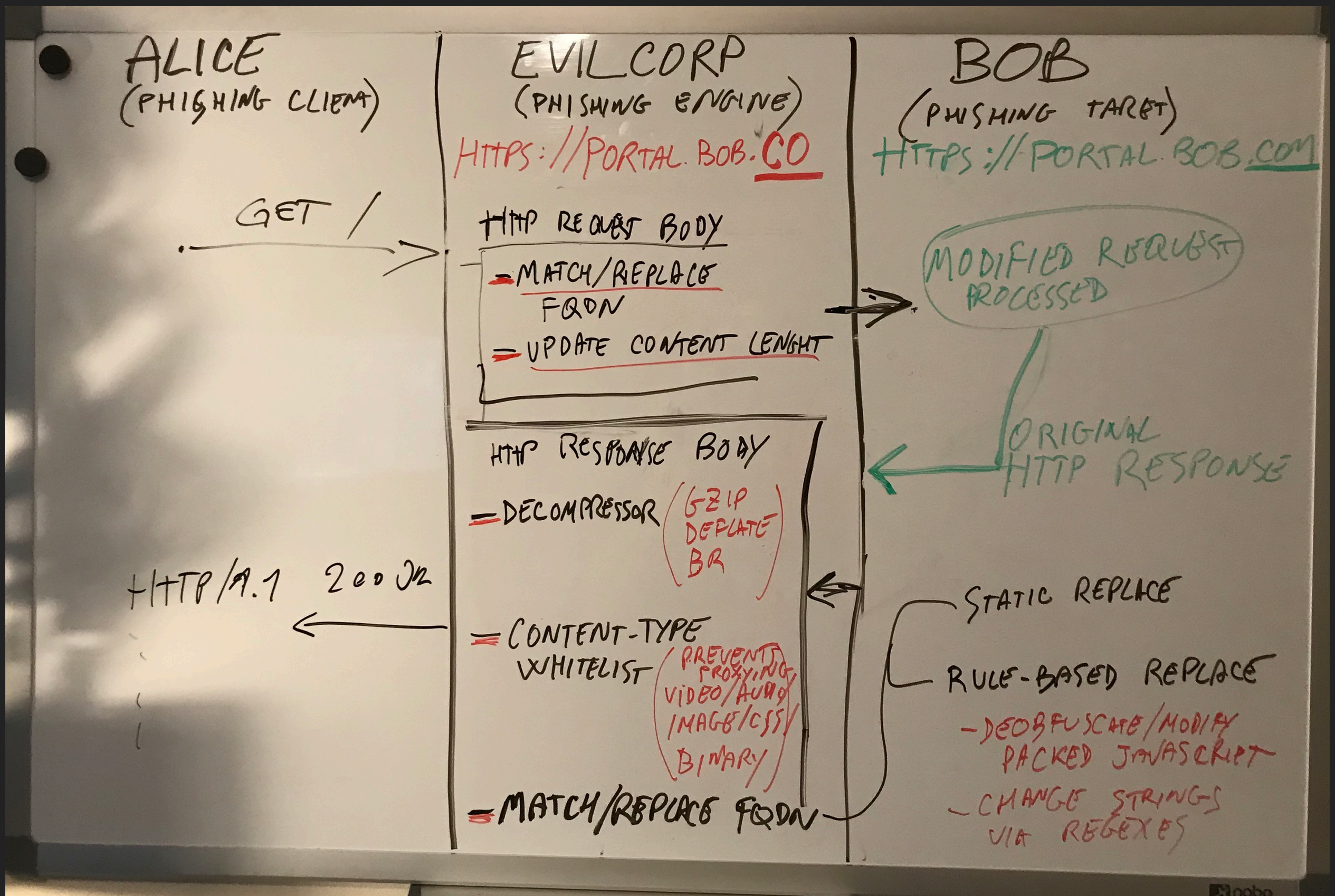


- How did the community react to this?

REVERSE PROXYING HEADERS



REVERSE PROXYING BODY



REVERSE PROXY: MURAENA GOLANG IMPLEMENTATION

- We choose Golang for:
  - High performance, great syntax
  - Stable core and good library ecosystem
  - Cross-compilation
- cloc . --exclude-dir vendor: ~2300 LoC

REVERSE PROXY: MURAENA GOLANG IMPLEMENTATION

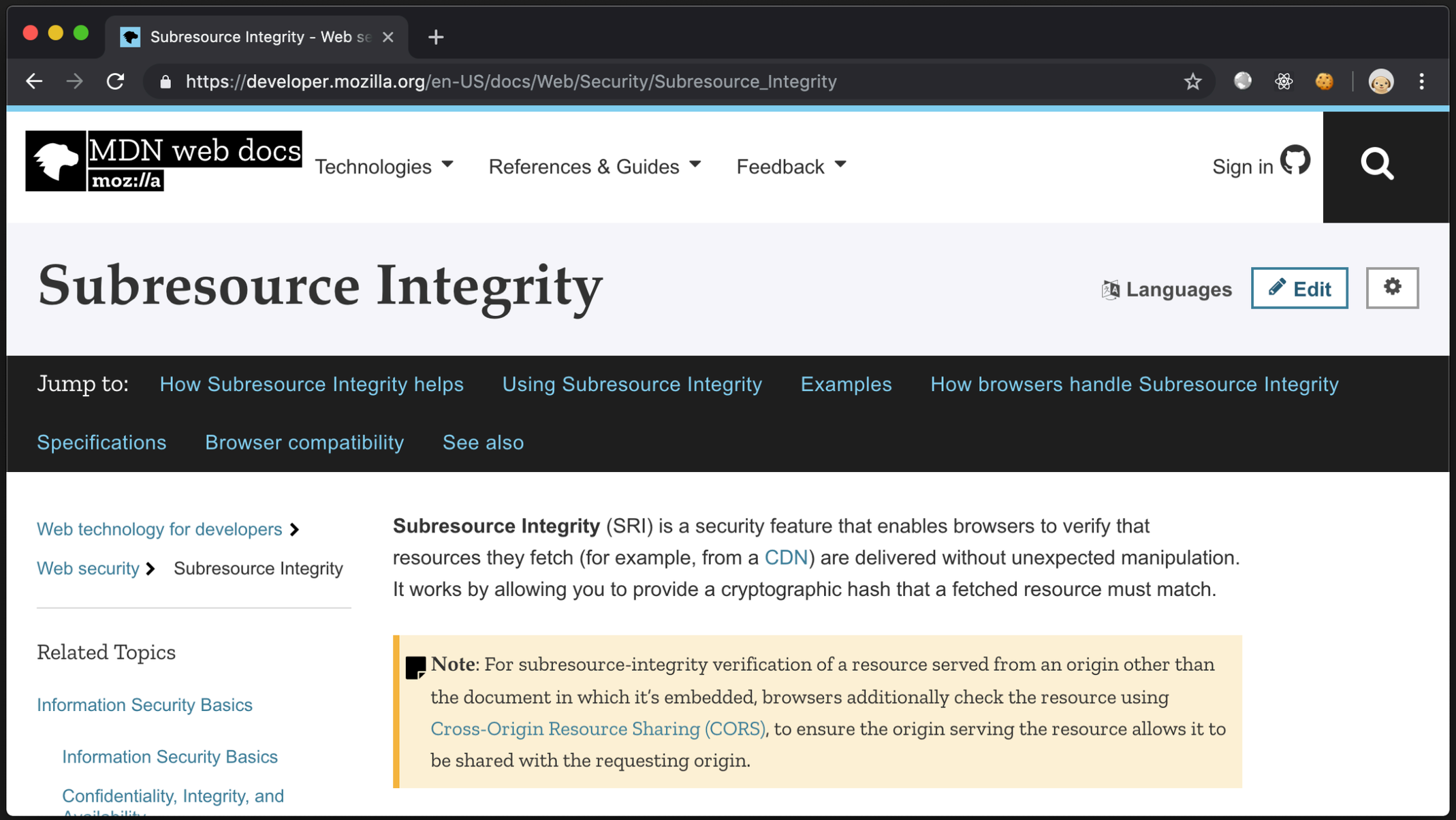
- Muraena comes with a number of unique features
  - Web Crawler
  - Customisable Tracking
  - Static Content Server
  - Wildcard domain support
  - Browser Instrumentation Integration
- The proxy core is net/http SingleHostReverseProxy with a custom transformer to grep & replace strings in HTTP request/response

REVERSING ANTI-REVERSE PROXY

- Some origins implement additional checks aimed at preventing proxying or framing
- Mostly easy checks, but reversing could get complicated with polymorphic heavily-obfuscated Javascript
- Chrome DevTools and Burp Proxy are your friends ;-)

REVERSING ANTI-REVERSE PROXY

- Universal manipulations



REVERSING ANTI-REVERSE PROXY

- Universal manipulations

```
<script src="https://example.com/example-framework.js"
        integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/
uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"
        crossorigin="anonymous"></script>
```

becomes

```
<script src="https://example.com/example-framework.js"
        no-integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K
uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"
        crossorigin="anonymous"></script>
```

REVERSING ANTI-REVERSE PROXY

- Universal manipulations
- Content-Security Policy annihilation!

```
<script src="https://example.com/example-framework.js"
        nonce="oqVuAfXRKap7fdgc"></script>
```

becomes

```
<script src="https://example.com/example-framework.js"
        no-more-nonce="oqVuAfXRKap7fdgc"></script>
```

REVERSING ANTI-REVERSE PROXY

- Universal manipulations
- Content-Security Policy annihilation!

```
<meta http-equiv="Content-Security-Policy" content="...">
```

becomes

```
<meta no-more-CSP content="...">
```

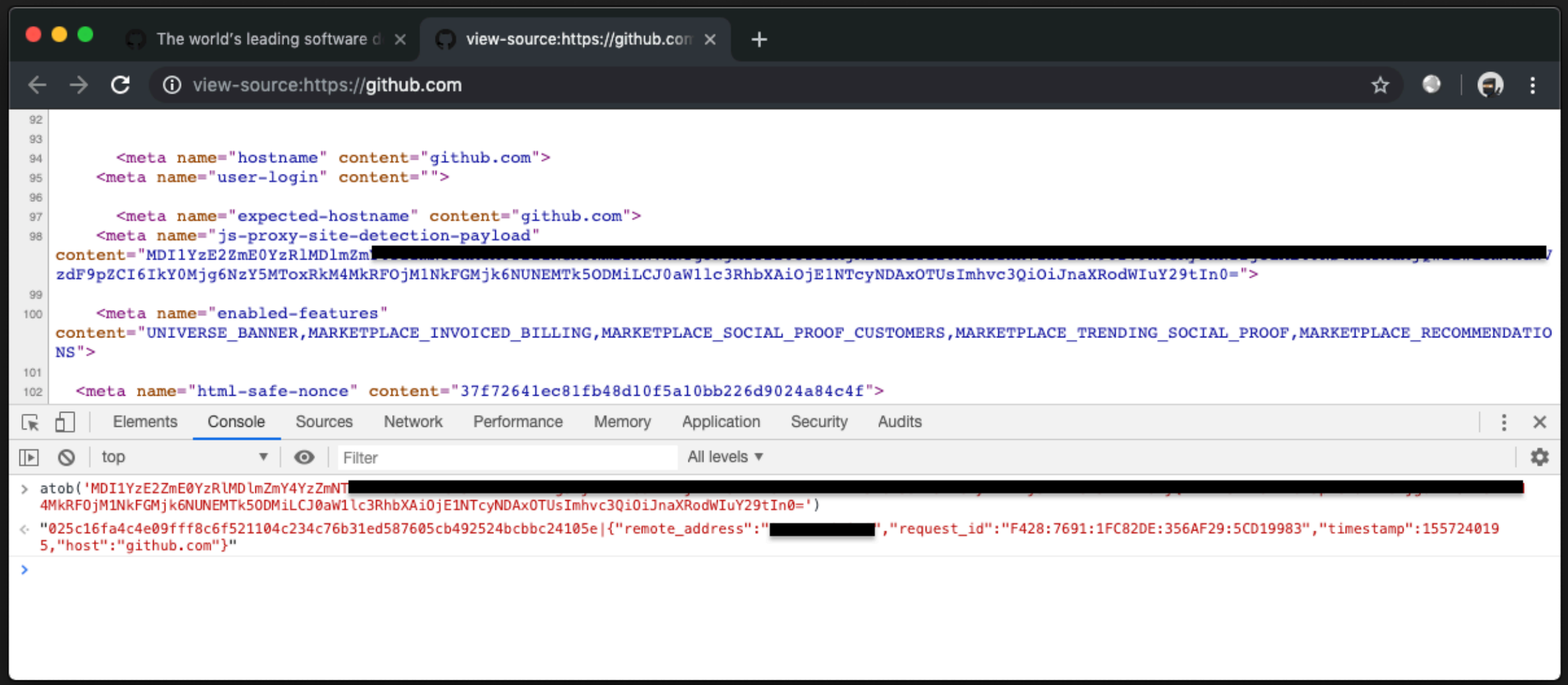
REVERSING ANTI-REVERSE PROXY

- Universal manipulations
- Content-Security Policy annihilation!

```
"remove": {
    ...,
    "response": {
      "header": [
        "Content-Security-Policy",
        "Content-Security-Policy-Report-Only",
        ...
      ]
    }
},
```

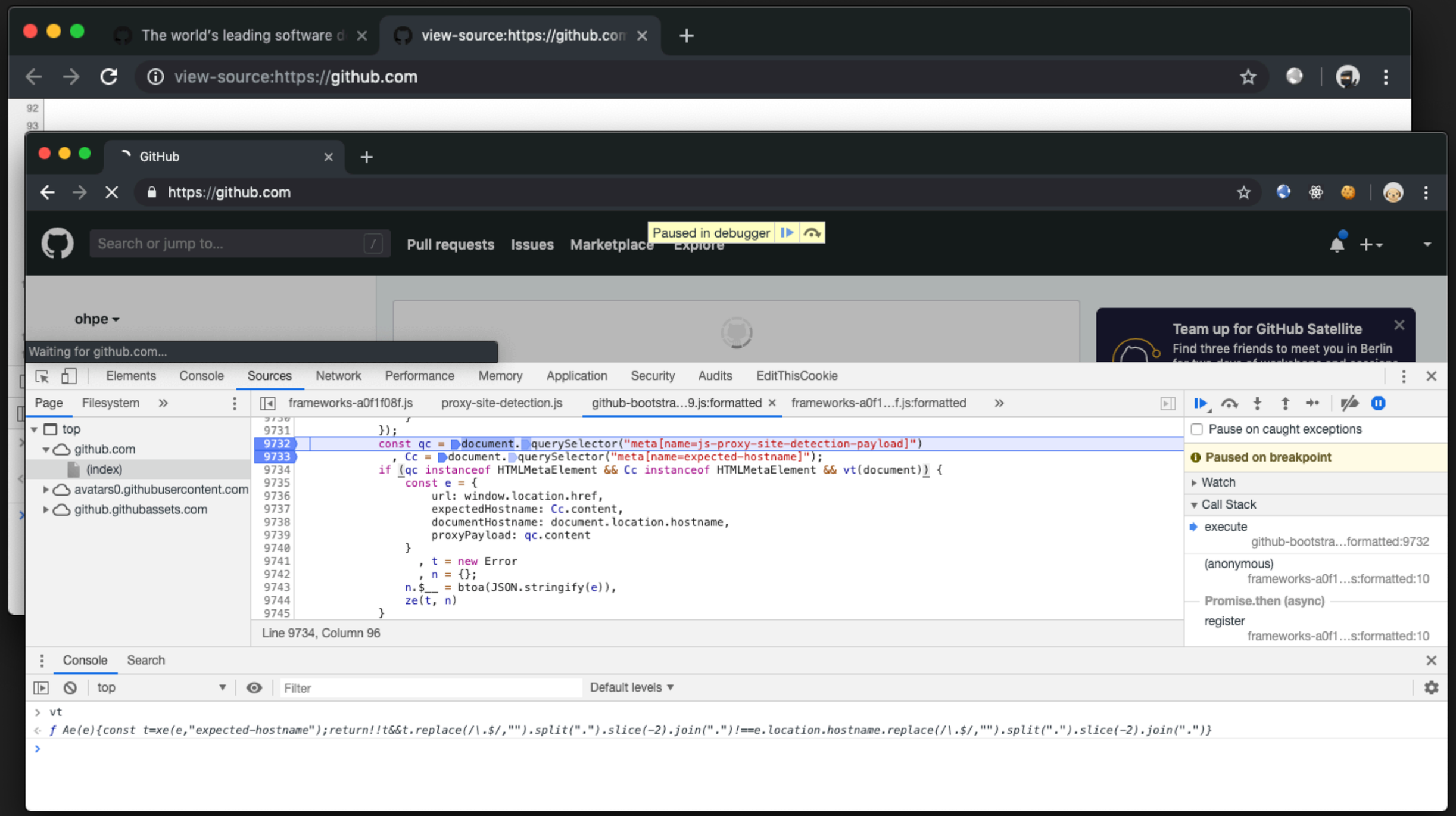
REVERSING ANTI-REVERSE PROXY: GITHUB

- An easy one: GitHub



REVERSING ANTI-REVERSE PROXY: GITHUB

- An easy one: GitHub



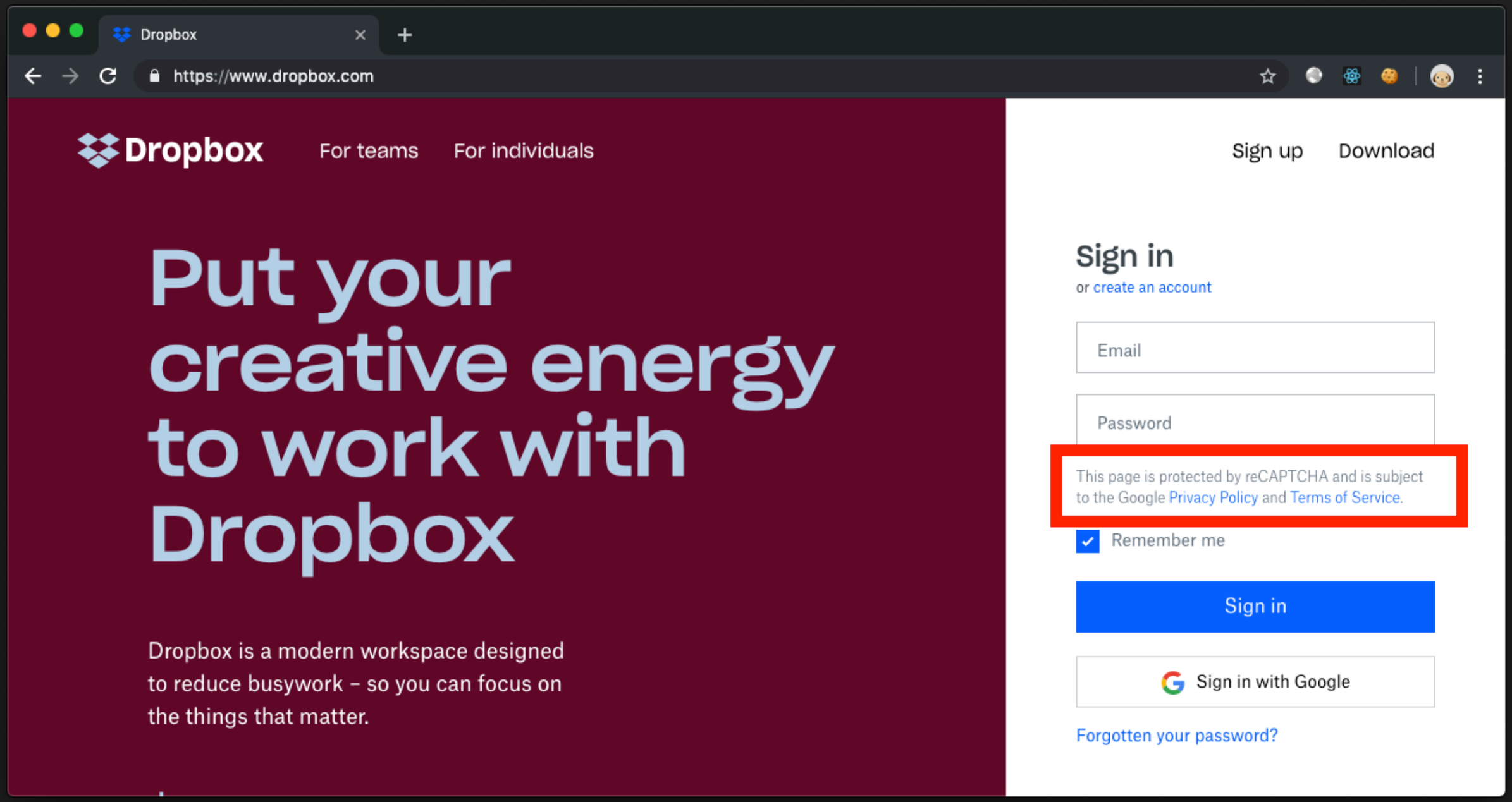
REVERSING ANTI-REVERSE PROXY: GITHUB

- An easy one: GitHub
- The fix: nullify the keywords:

```
"js-proxy-site-detection-payload" -> ""
"expected-hostname" -> ""
```

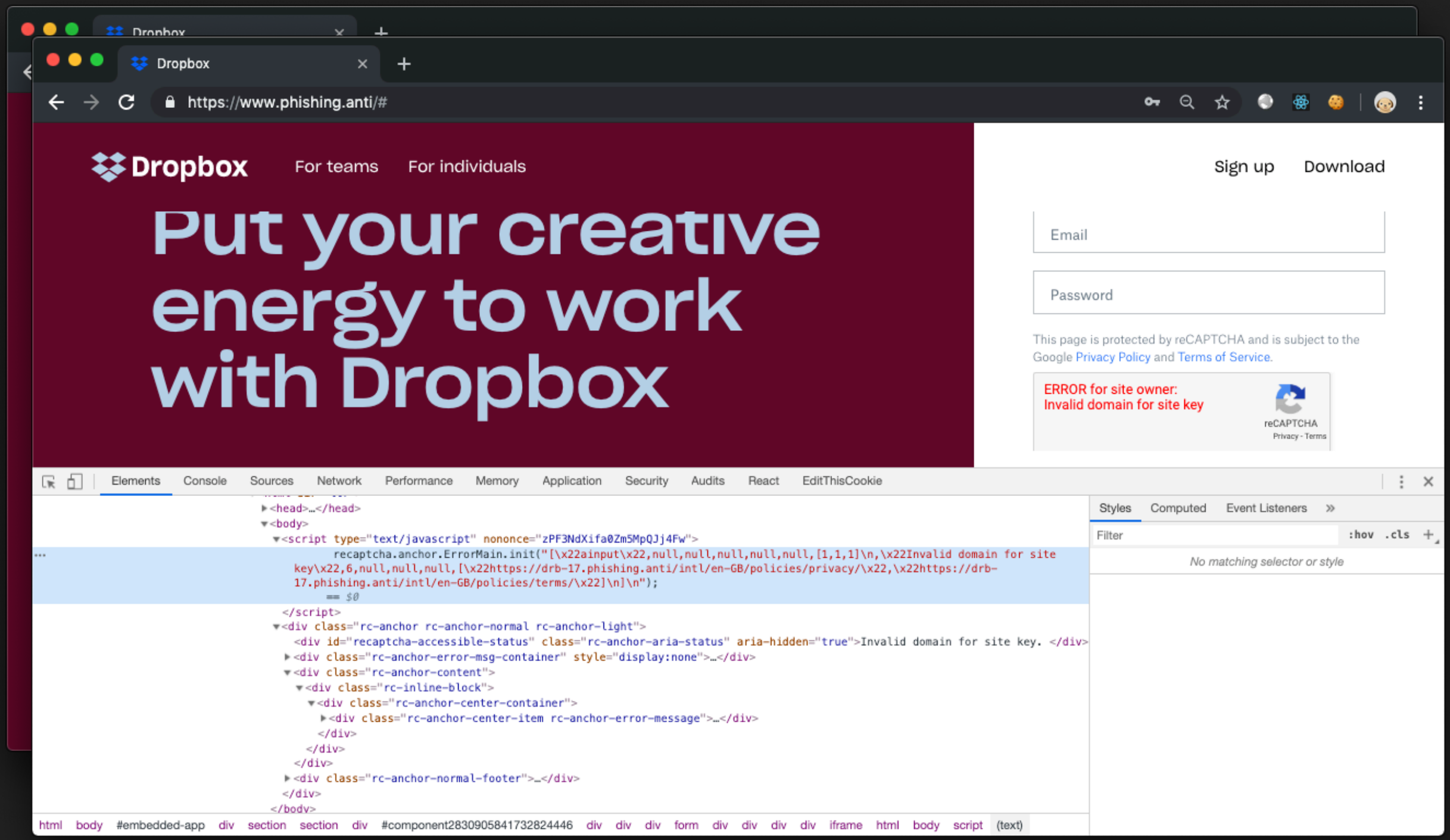
REVERSING ANTI-REVERSE PROXY: DROPBOX

- The reCAPTCHA challenge



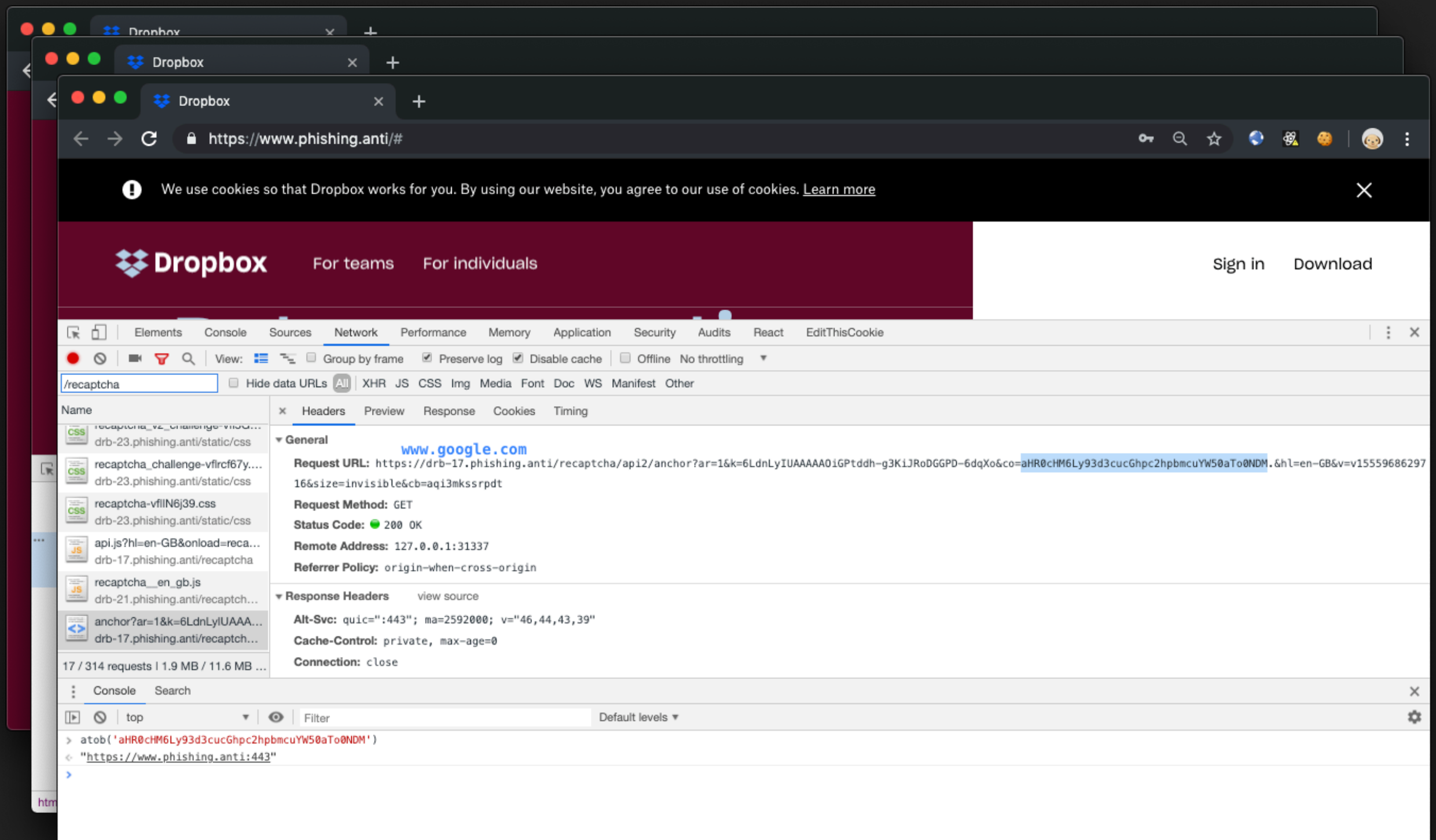
REVERSING ANTI-REVERSE PROXY: DROPBOX

- The reCAPTCHA challenge



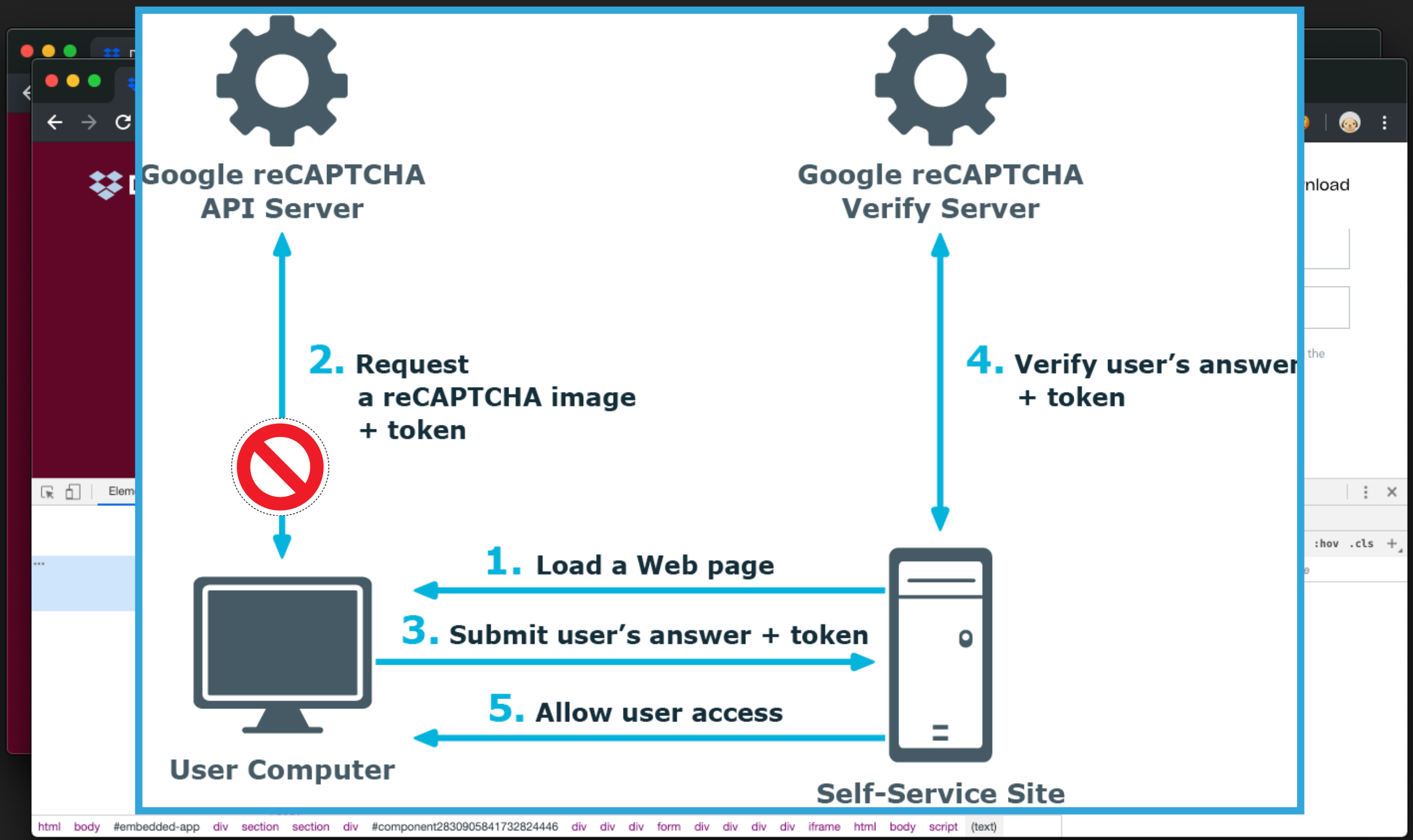
REVERSING ANTI-REVERSE PROXY: DROPBOX

- The reCAPTCHA challenge



REVERSING ANTI-REVERSE PROXY: DROPBOX

- The reCAPTCHA challenge



REVERSING ANTI-REVERSE PROXY: DROPBOX

- The reCAPTCHA challenge
- The fix: base64 transformation support:

```
"transform": {
        "base64": {
          "enabled": true,
    "padding": [
      "=",
      "."
    ]
  },
```

REVERSING ANTI-REVERSE PROXY: GSUITE

- If it works for Google works for all

REVERSING ANTI-REVERSE PROXY: GSUITE

- If it works for Google works for all
- There are several regexes to patch:

```
["|(google))\\.com","|(google)|(phishing))(\\.anti|\\.com)"],
[".google\\.(((co|com)",".(google|phishing)\\.(((co|com|anti)"],
[".google\\.com",".phishing\\.anti"],
["\\.google(rs)?\\.com","\\.(google(rs)|phishing)?\\.(com|anti)"],
["LCJwcHUiOiJodHRwczovL21haWwuZ29vZ2xlLmNvbS9yb2JvdHMudHh0IiwibHB1IjoiaHR0cHM6Ly9oYW5nb3V0cy5nb29nbGUuY29tL3JvYm90cy50eHQifQ",
"LCJwcHUiOiJodHRwczovL21haWwucGhpc2hpbmcuYW50aS9yb2JvdHMudHh0IiwibHB1IjoiaHR0cHM6Ly9oYW5nb3V0cy5waGlzaGluZy5hbnRpL3JvYm90cy50eHQifQ=="
],
["LCJwcHUiOiJodHRwczovL2hhbmdvdXRzLmdvb2dsZS5jb20vcm9ib3RzLnR4dCIsImxwdSI6Imh0dHBzOi8vaGFuZ291dHMuZ29vZ2xlLmNvbS9yb2JvdHMudHh0In0",
"LCJwcHUiOiJodHRwczovL2hhbmdvdXRzLnBoaXNoaW5nLmFudGkvcm9ib3RzLnR4dCIsImxwdSI6Imh0dHBzOi8vaGFuZ291dHMucGhpc2hpbmcuYW50aS9yb2JvdHMudHh0In0="]
```

THE ART OF INSTRUMENTING STOLEN WEB SESSIONS

- Since all the traffic is passing through Muraena, credentials and session cookies are captured

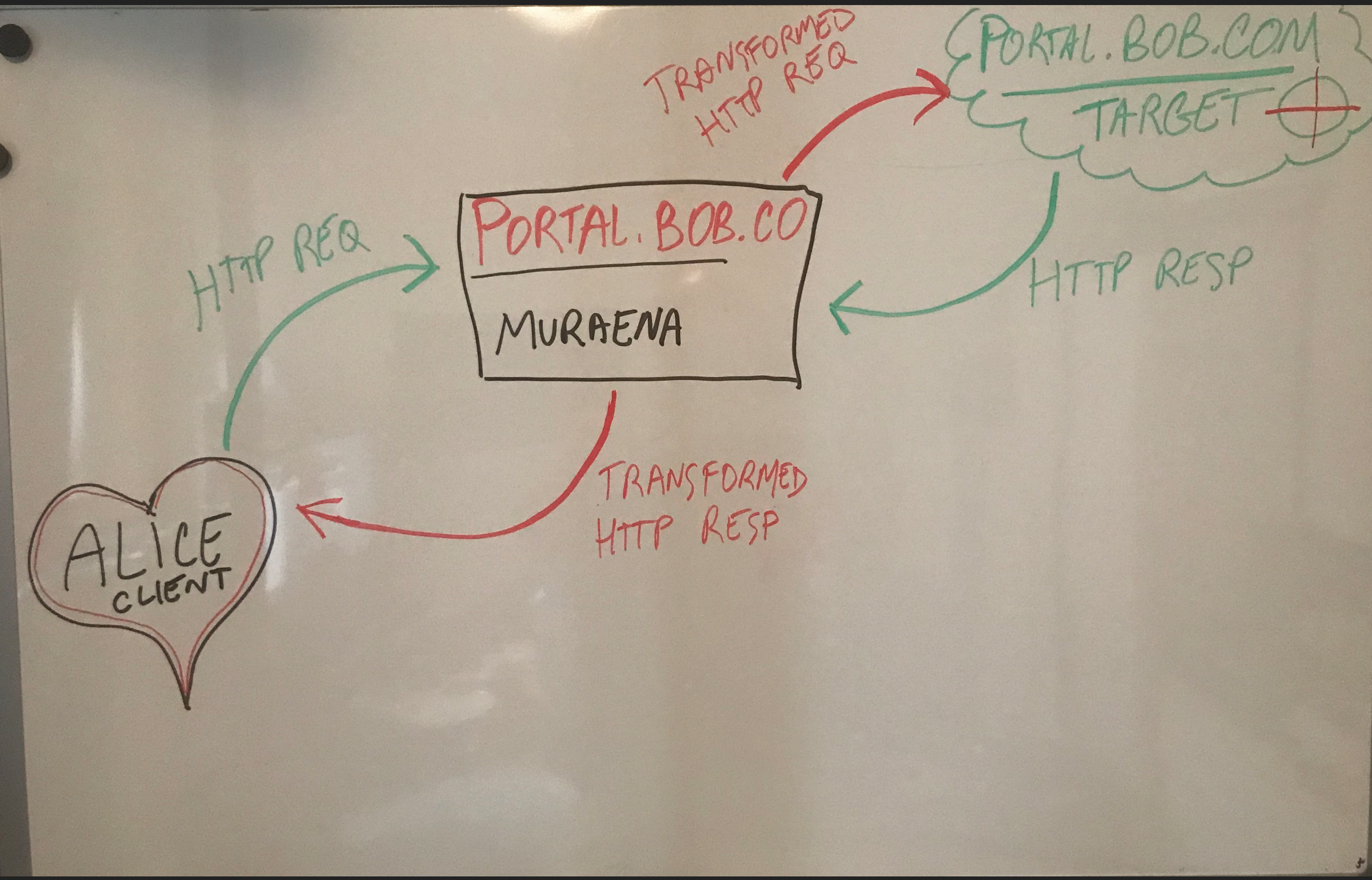
- Is the targeted origin able to spot if we hijack the authenticated session passing it to an instrumented browser?

THE ART OF INSTRUMENTING STOLEN WEB SESSIONS

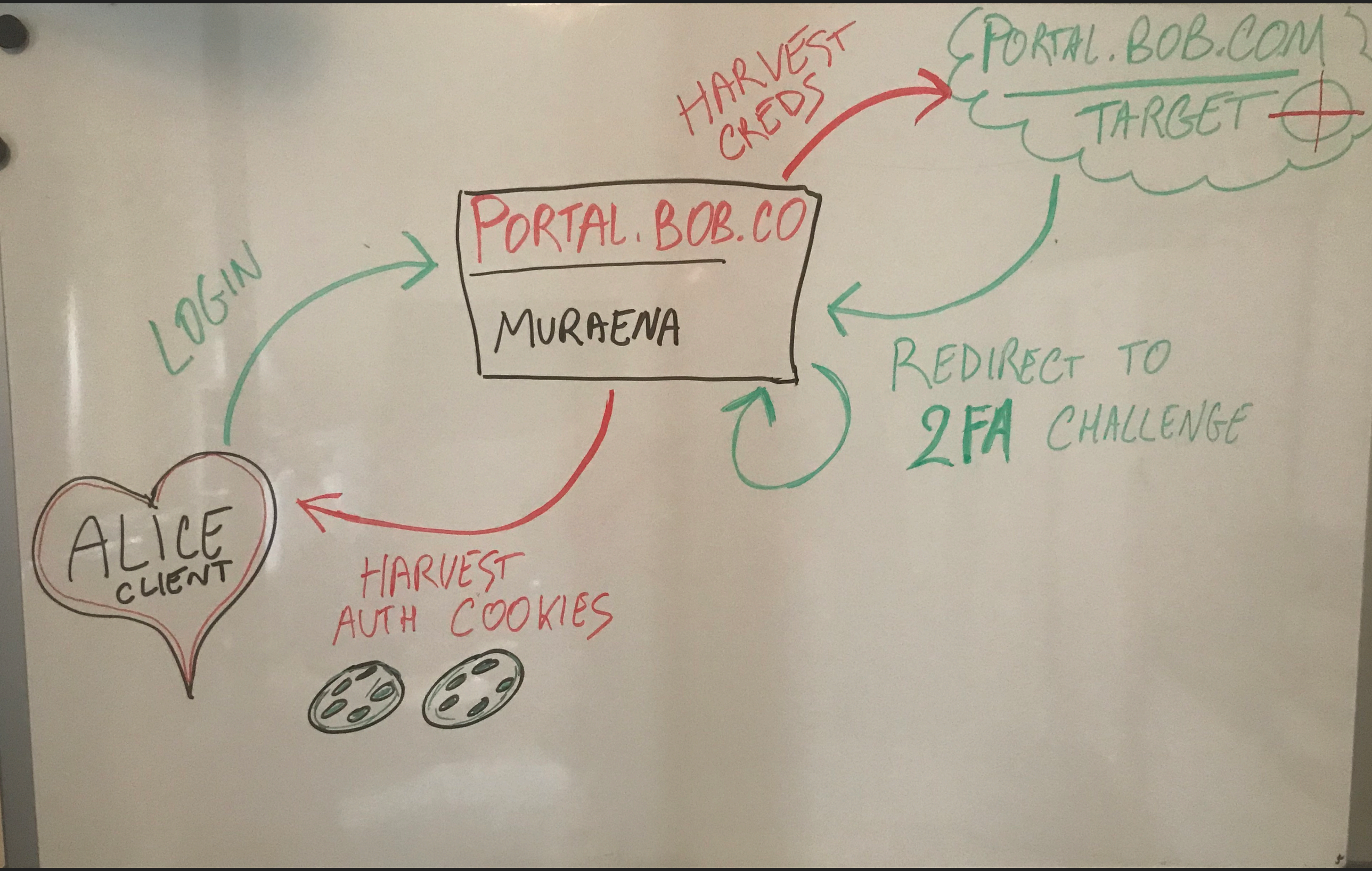
- Since all the traffic is passing through Muraena, credentials and session cookies are captured

- Is the targeted origin able to spot if we hijack the authenticated session passing it to an instrumented browser?
  - Usually **NO**
  - Additionally: the instrumented browser connection goes out via the same IP of Muraena, and the UA is changed to reflect the victim one.

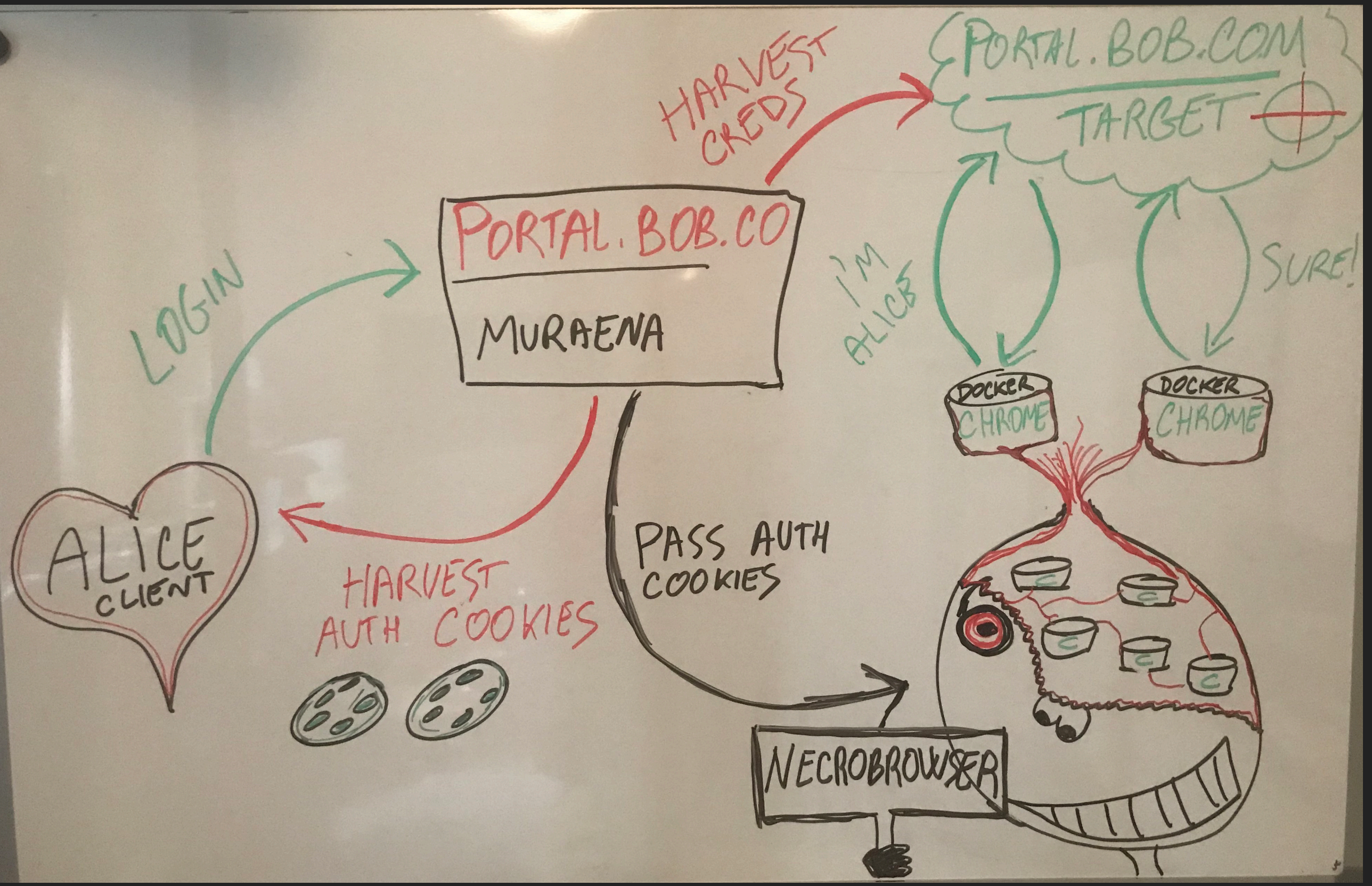
MURAENA WITH NECROBROWSER



MURAENA WITH NECROBROWSER



MURAENA WITH NECROBROWSER



NECROBROWSER

- NecroBrowser is a wrapper around chromedp (https://github.com/chromedp/chromedp)
  - Programmatically drive Chrome via Chrome DevTools Protocol (CDP)
  - Exposed as a micro service that spawns dedicated Docker containers with Chrome
- Allows to keep alive as many session as your Docker server/cluster can support
- Can be scheduled to do repeated actions (dump emails every hour, read Slack messages every minute)

HEADLESS CHROME DETECTION

http://antoinevastel.github.io/bot%20detection/2018/01/17/detect-chrome-headless-v2.html

https://intoli.com/blog/not-possible-to-block-chrome-headless/

Current status:

```
Headless detection *failed*.
😎   Evaders are winning!
```

https://github.com/paulirish/headless-cat-n-mouse

HEADLESS CHROME DETECTION

- https://intoli.com/blog/making-chrome-headless-undetectable/
- Simply, it's not easy to detect a non-human driven browser

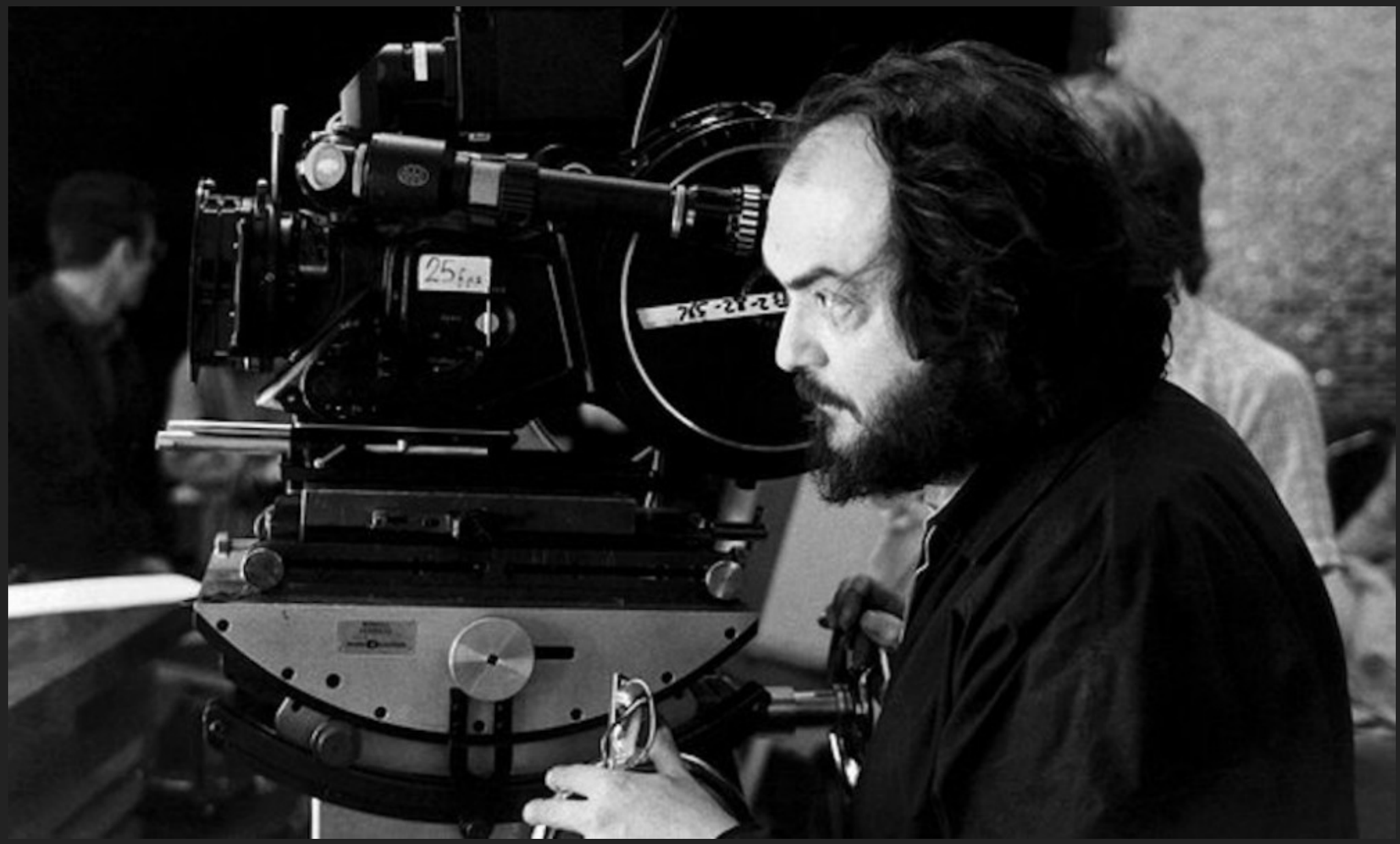
Test Name		Result
User Agent	(Old)	Mozilla/5.0 (X11; Linux x86_64)AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.39 Safari/537.36
WebDriver	(New)	missing (passed)
Chrome	(New)	present (passed)
Permissions	(New)	pending
Plugins Length	(Old)	5
Languages	(Old)	en-US,en

AUTOMATING POST-PHISHING ACTIVITIES

- Examples could be endless, but the following are already implemented:
  - Disable GitHub Notifications, add SSH Key, download all repositories code
  - Disable GSuite Notifications, add Application Password, Dump Email
- Upload arbitrary files to Dropbox, GDrive, Confluence
- Screenshot and HTML dump all the things

FULL-CHAIN DEMOS

- Come se fosse Antani col video registrato anche per lei, senno' son moccoli anche se non partissero come ieri?



UPCOMING CHALLENGES

- Google:

Better protection against Man in the Middle phishing attacks

April 18, 2019

However, one form of phishing, known as "man in the middle" (MITM), is hard to detect when an embedded browser framework (e.g., Chromium Embedded Framework - CEF) or another automation platform is being used for authentication. MITM intercepts the communications between a user and Google in real-time to gather the user's credentials (including the second factor in some cases) and sign in. Because we can't differentiate between a legitimate sign in and a MITM attack on these platforms, we will be blocking sign-ins from embedded browser frameworks starting in June. This is similar to the restriction on webview sign-ins announced in April 2016.

What developers need to know

The solution for developers currently using CEF for authentication is the same: browser-based OAuth authentication. Aside from being secure, it also enables users to see the full URL of the page where they are entering their credentials, reinforcing good anti-phishing practices. If you are a developer with an app that requires access to Google Account data, switch to using browser-based OAuth authentication today.

FUTURE WORK

- Bettercap integration 
- More browser automation fun:
  - Support for more commonly used web portals
  - Scaling tests, queuing instrumentation jobs
- Use browser instrumentation also for RECON/OSINT pre-phishing:
  - Scrape company X profile from LinkedIn/SocialNetworks using a real browser with a fake account

WHERE TO FIND THE CODE

https://github.com/muraenateam

THANKS!

QUESTIONS?