

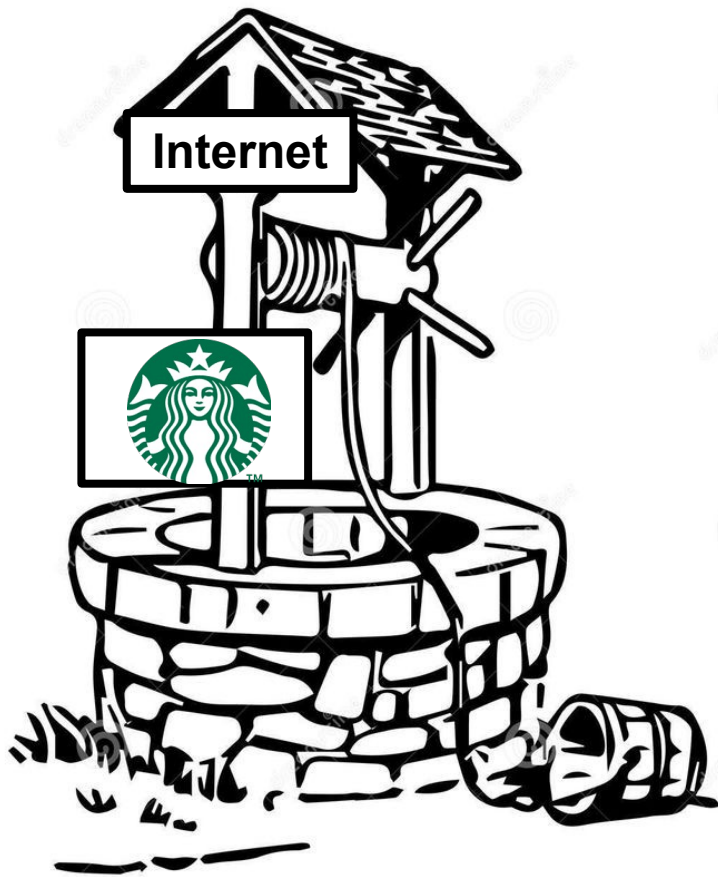
HSTS and Cookies: Side-channels to Steal Browsing History

Adrian Dabrowski
adabrowski@sba-research.org
Twitter: @atrox_at

Co-Authors: Georg Merzdovnik, Nikolaus Kommenda, Edgar R. Weippl

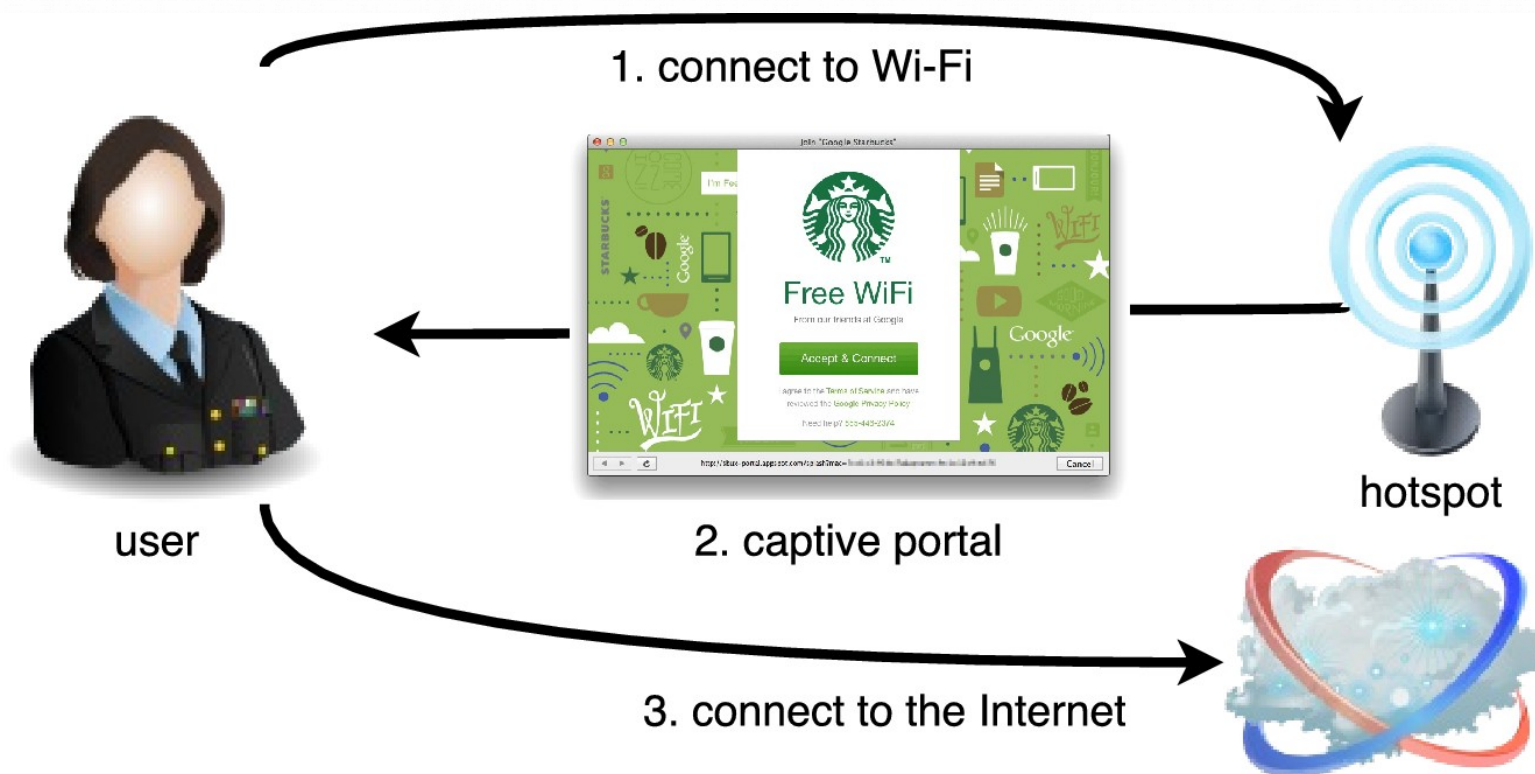
DeepSec 2016-11-10

Public Wi-Fi Hotspots



- Like a well in a village
- We gather there, pull up a bucket or two of “Internet”
- Look at the sign from the sponsor
- ... and move on.

What is a “Captive Portal”?



Why Captive Portal

- Omnipresent in Wi-Fi Hotspots
 - Used by you probably right now (in this very hotel)
- Has an elevated position on the network
- Man-in-the-Middle by design
 - Sponsors of a Wi-Fi want us to see their messages (and accept the disclaimer)
 - There is no standard for that
 - Let's inject it into your traffic...



sergey bratus

@sergeybratus



Following

As more sites go HTTPS & more Wi-Fi goes captive portal, I find myself treasuring short names of plain old HTTP sites that get MITMed faster

RETWEETS

3

LIKES

4



Browser History Stealing, again?

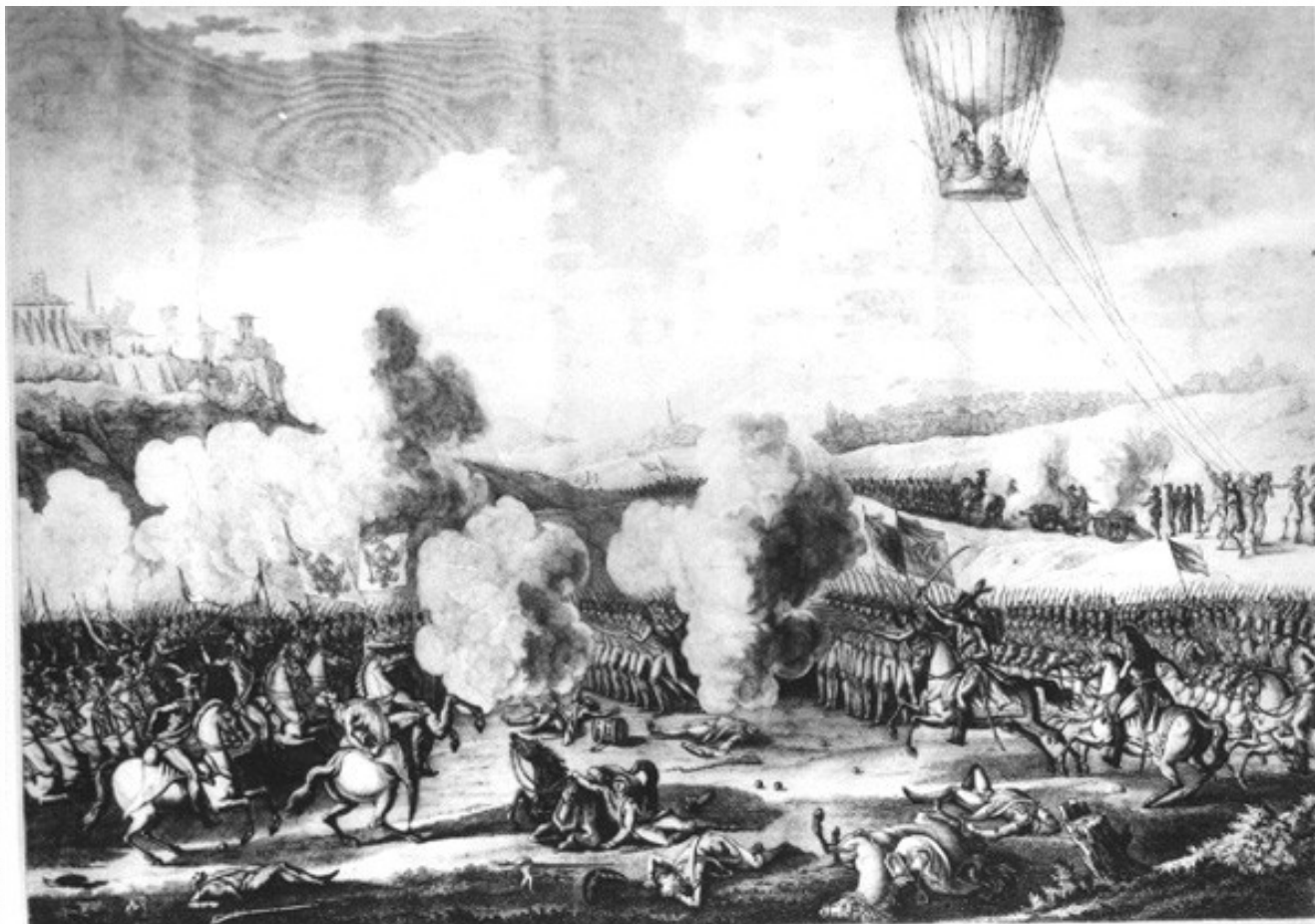
- Baron, 2002
 - :visited link color
- Ruderman, 2000
 - :visited can load images
- Jang, 2010
 - Sites are actively trying to steal history



History, so what?

- Culture & Language
 - Amazon.fr, Amazon.jp
 - Sexual orientation
 - grindr.com, transblog.de
 - Partnership status
 - Okcupid.com, parship.com
 - Employer
 - intranet.ibm.com
- Other websites that give interesting insights
 - Medical conditions
 - Political campaigns
 - Religious communities

Reconnaissance

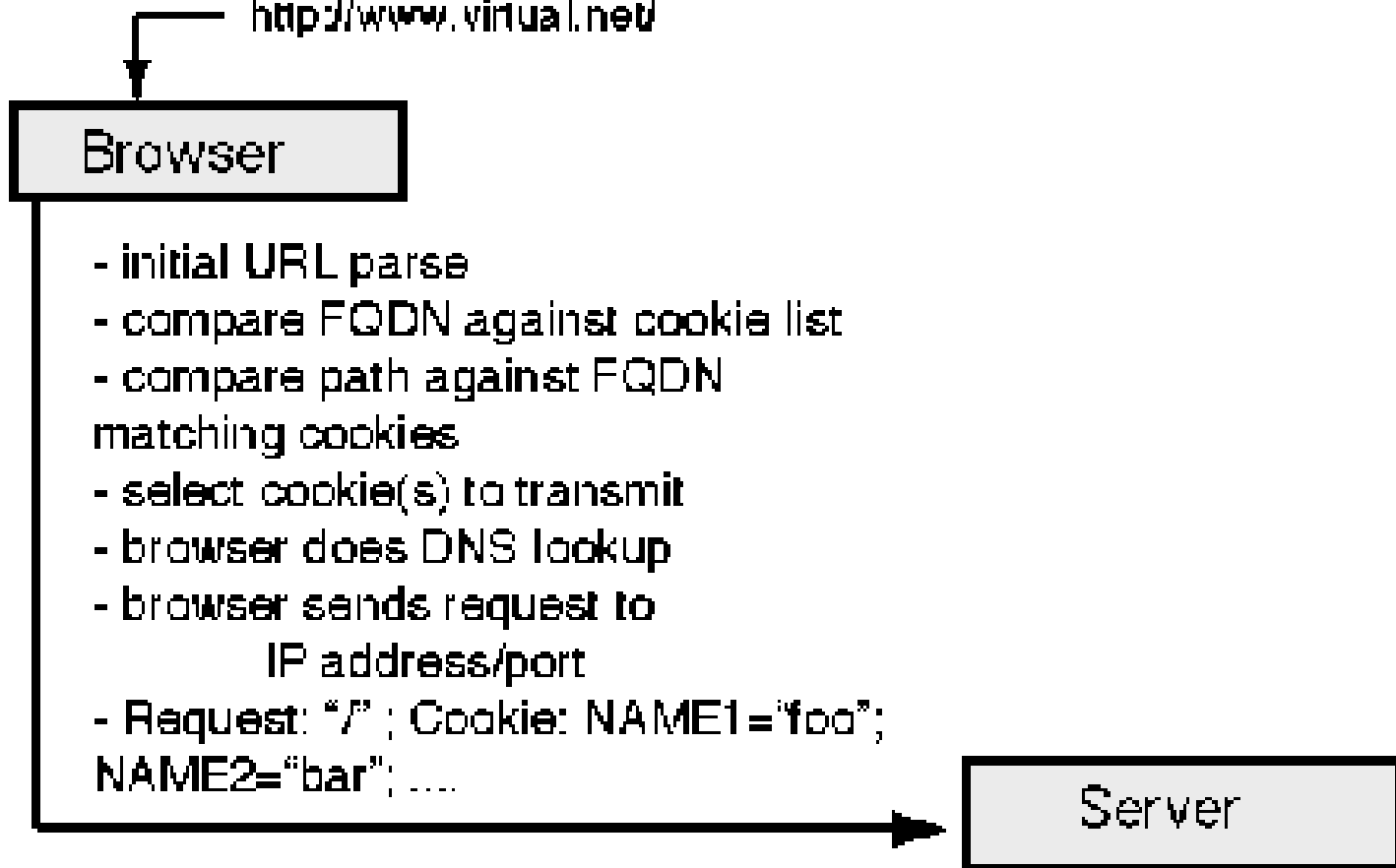


Bataille de Fleurus gagnée par l'Armée Française, le 8 Messidor, de l'An 2.

HTTP Request w/Cookie

URL is entered by user

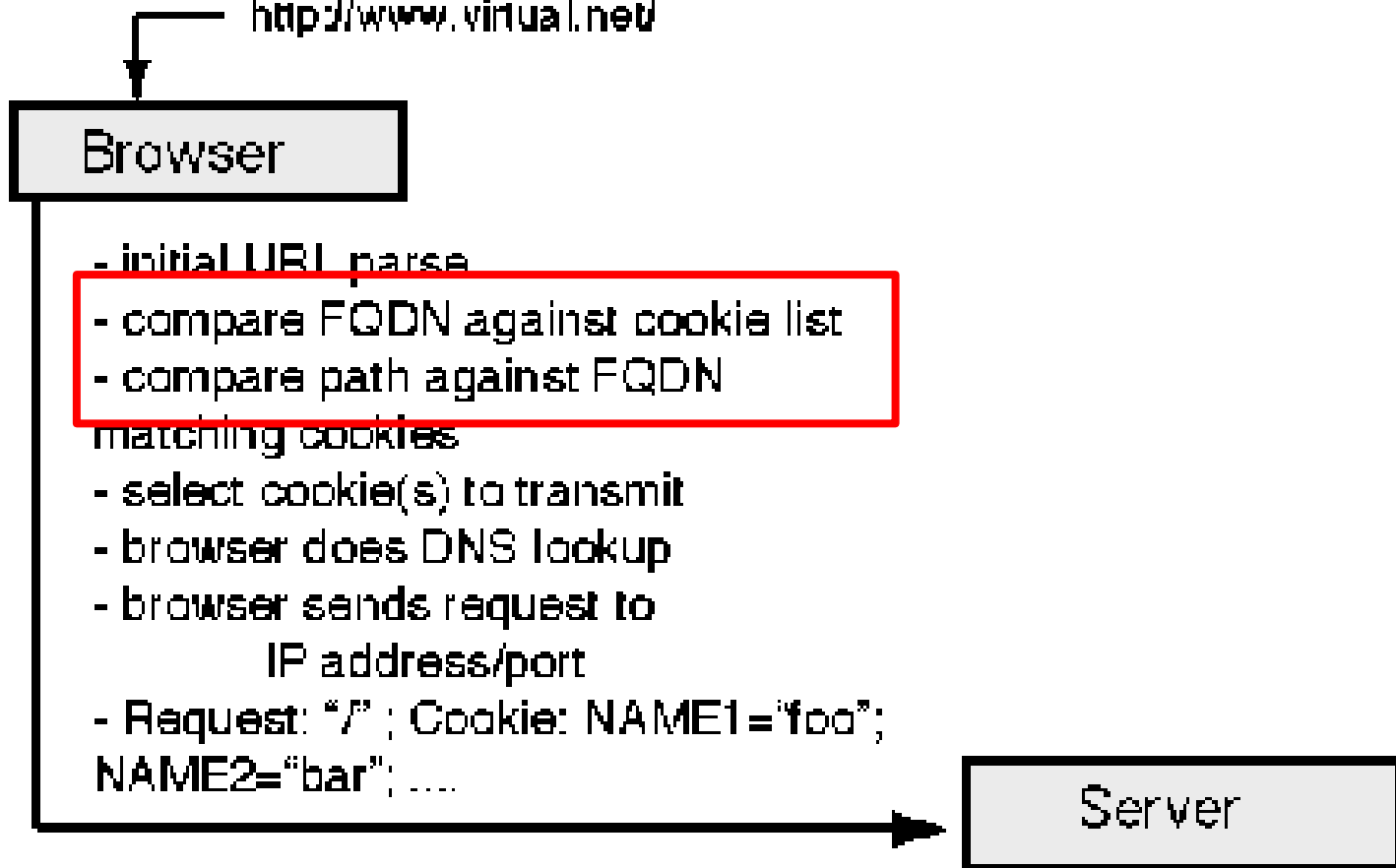
`http://www.virtual.net/`



HTTP Request w/Cookie

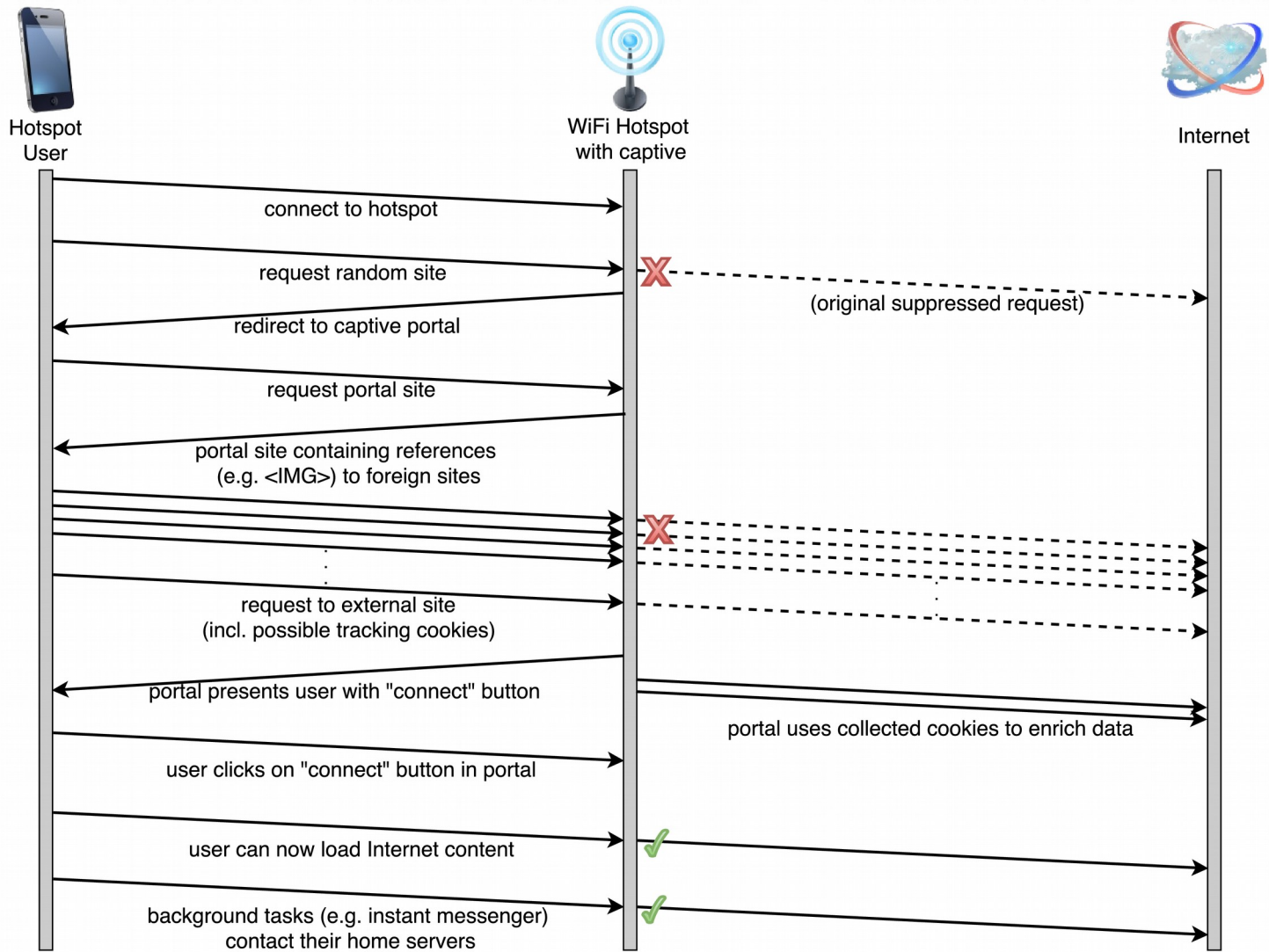
URL is entered by user

http://www.virtual.net/



Cookies (or not enough state for HTTP)

- Two kinds
 - Session cookies: usually forgotten when browser closed
 - Persistent cookies: stored on disk with expiry date
- Only depend on the FQDN and Protocol
 - XSS
 - XSRF
 - HTTP set cookie also used for HTTPS
 - Insecure set cookies mixed into the cookies over HTTPS



Hotspot User

WiFi Hotspot with captive portal

http://cnn.com (+cookies)

connect to hotspot

request random site

**302 redirect
login.hotspotsys.com/login**

redirect to captive portal

login.hotspotsys.com/login

request portal site

**<html>.... <img href=
"a.com/probe123"><img
href="b.com/probe123"> ..**

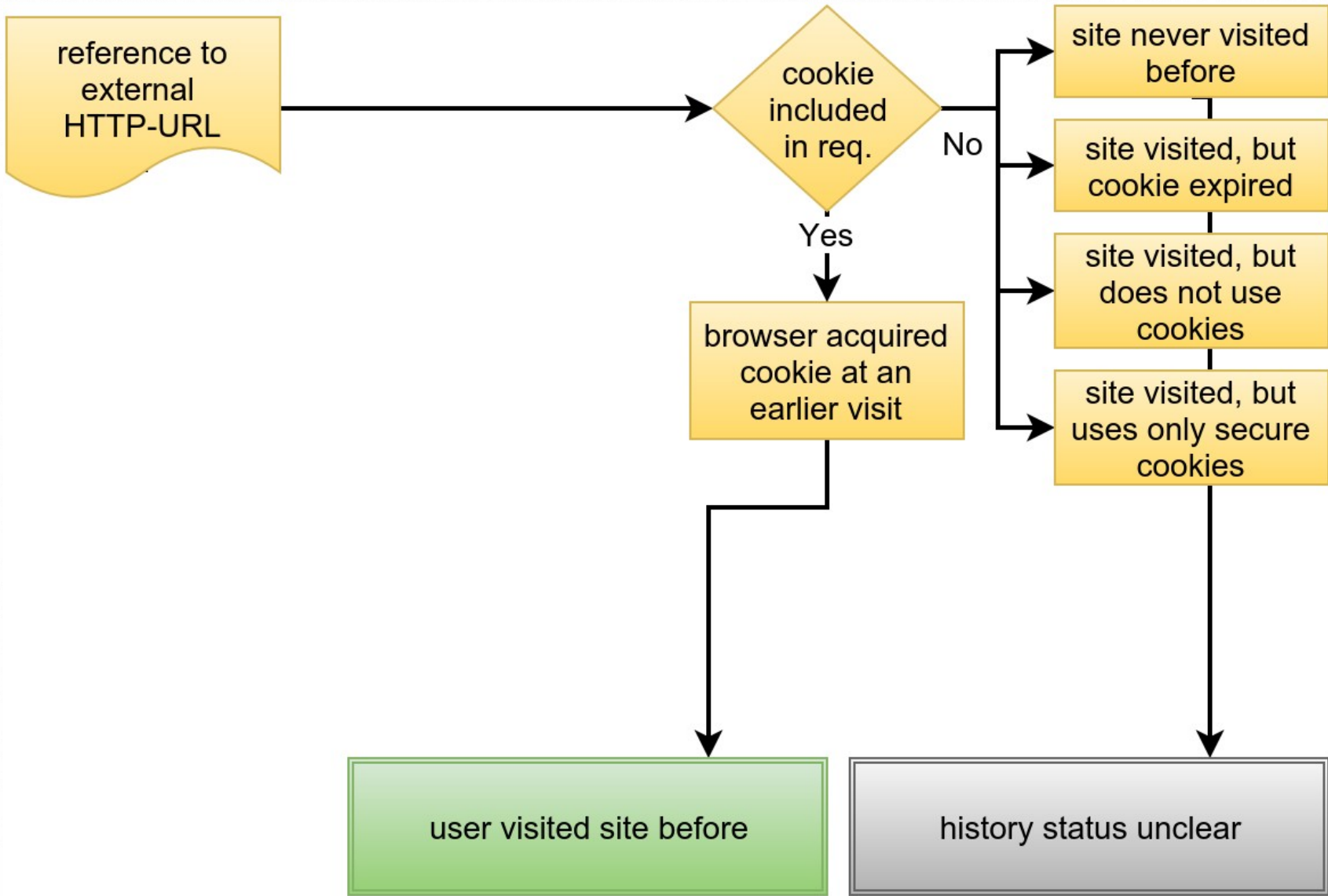
portal site containing references
(e.g.) to foreign sites

http://a.com/probe123 (+cookies)

http://b.com/probe123 (+cookies)

....

request to external site
(incl. possible tracking cookies)

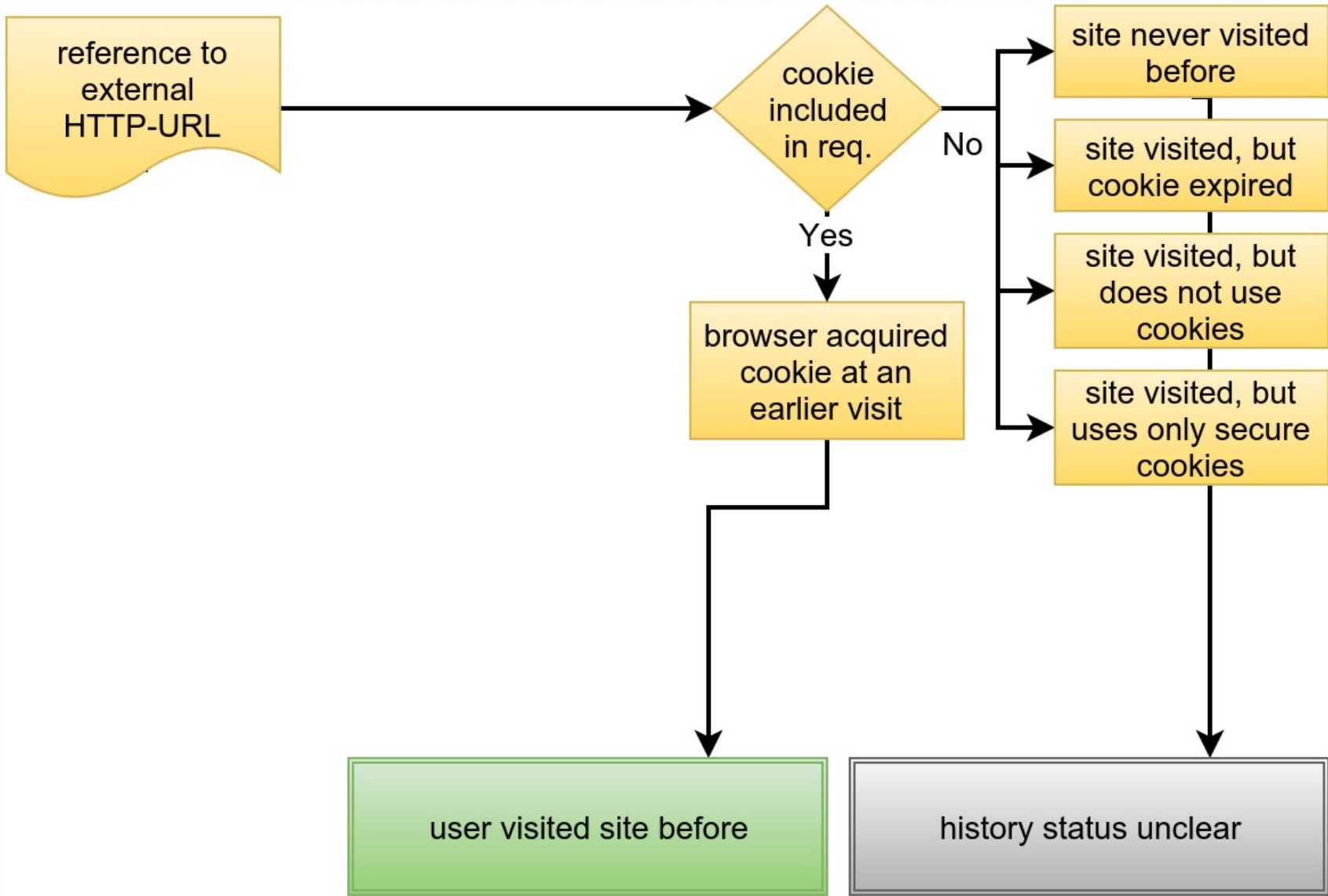


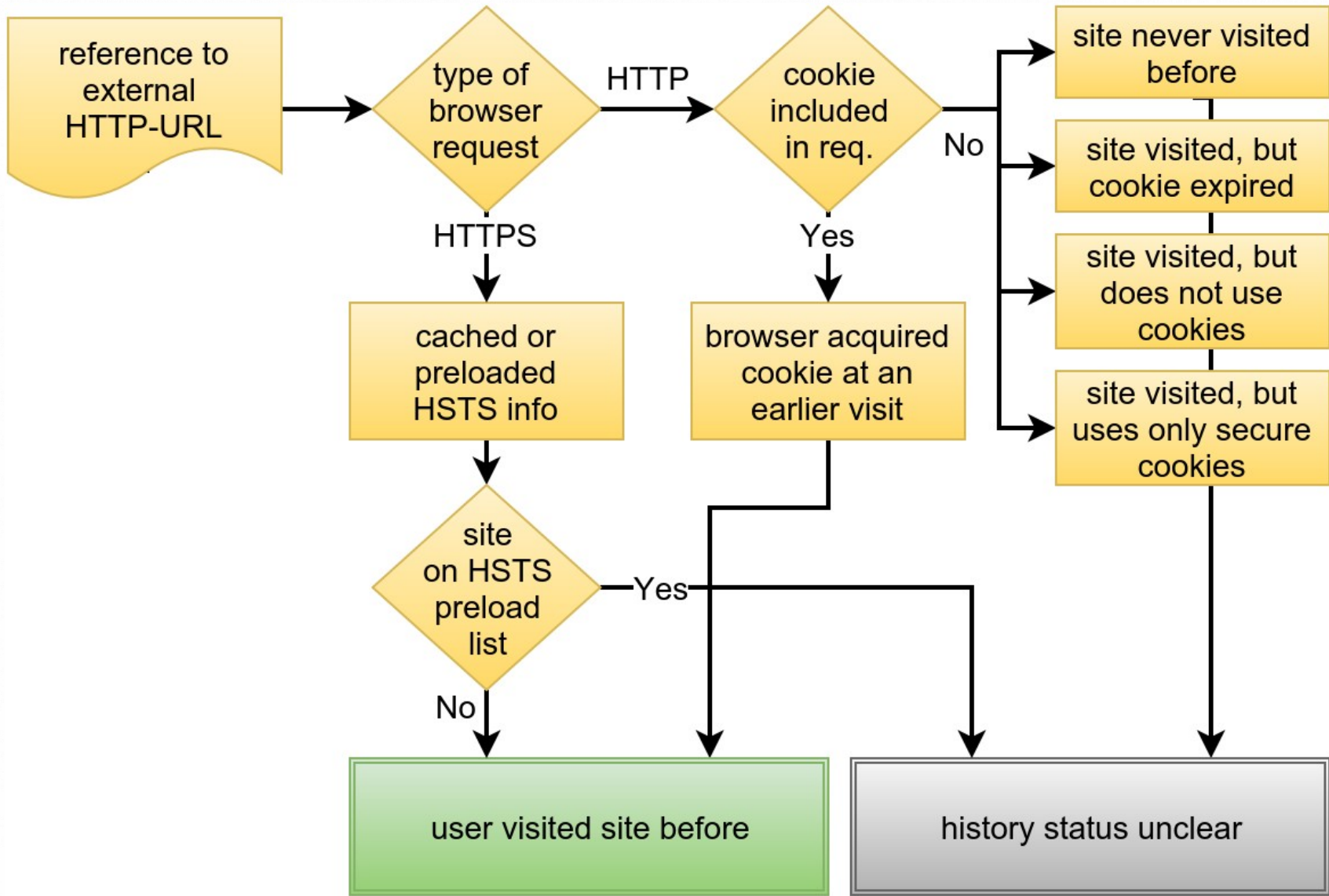
Sure, crypto will save us!

- HTTPS hides content (and therefore cookies)
 - But is not used by default (e.g., when URL entered in location bar)
- HTTP Strict Transport Security (HSTS)
 - Site announces availability of HTTPS with same content as via HTTP
 - Client caches this
 - Uses HTTPS by default next time for this site

Sure, crypto will save us!

- HTTPS hides traffic (and cookies)
 - But is not used by default (e.g., when URL entered in location bar)
- HTTP Strict Transport Security (HSTS)
 - Site announces availability of HTTPS with same content as via HTTP
 - Client caches this **side channel**
 - Uses HTTPS by default next time for this site



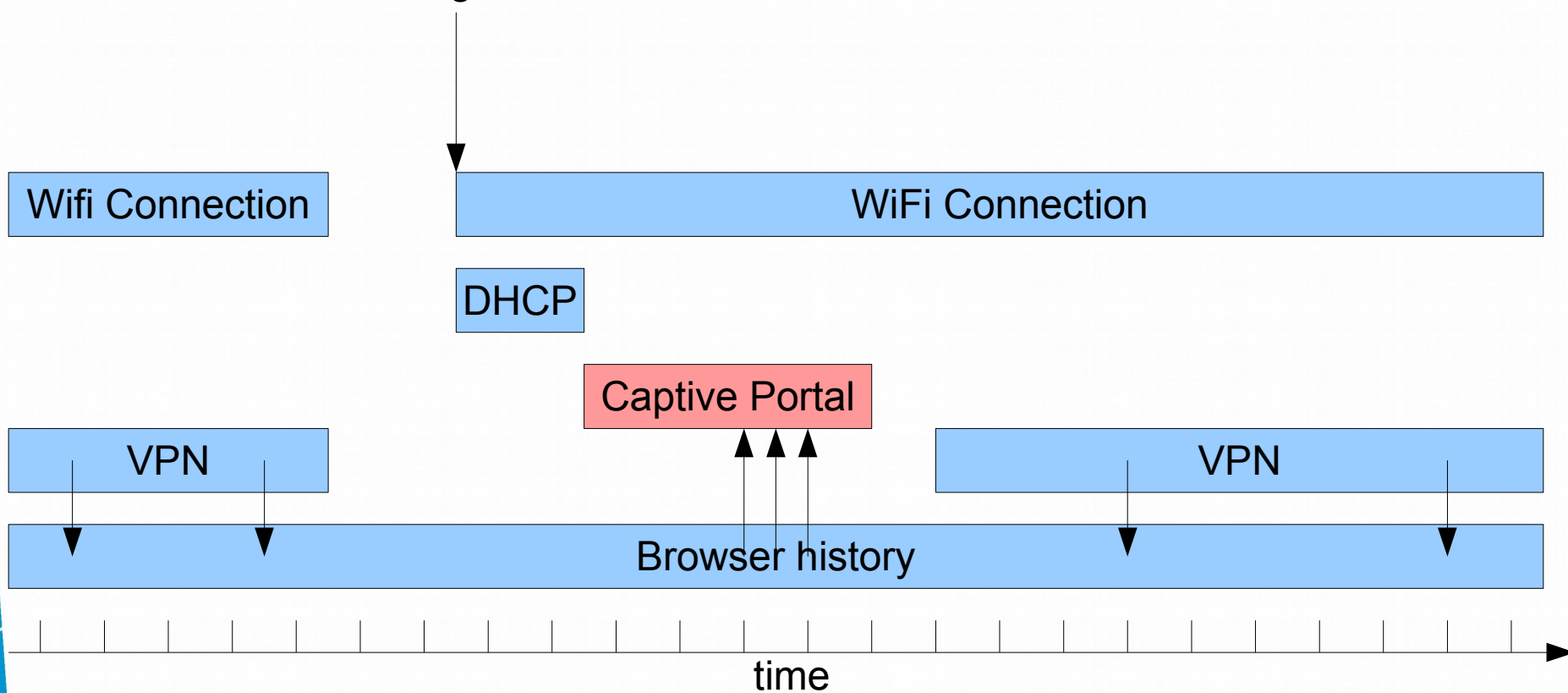


Which users are affected?

- Everyone who uses the standard browser to login into the captive portal (mobile, notebook, ...)
 - Even VPN users
- Android and iOS introduced captive portal detectors
 - Primarily for convenience – starts stripped down browser
 - The online test is very easy to fool, since based on HTTP
 - User will use main browser to login, exposing their history

Sure, VPN will save us?

Connecting to new WiFi



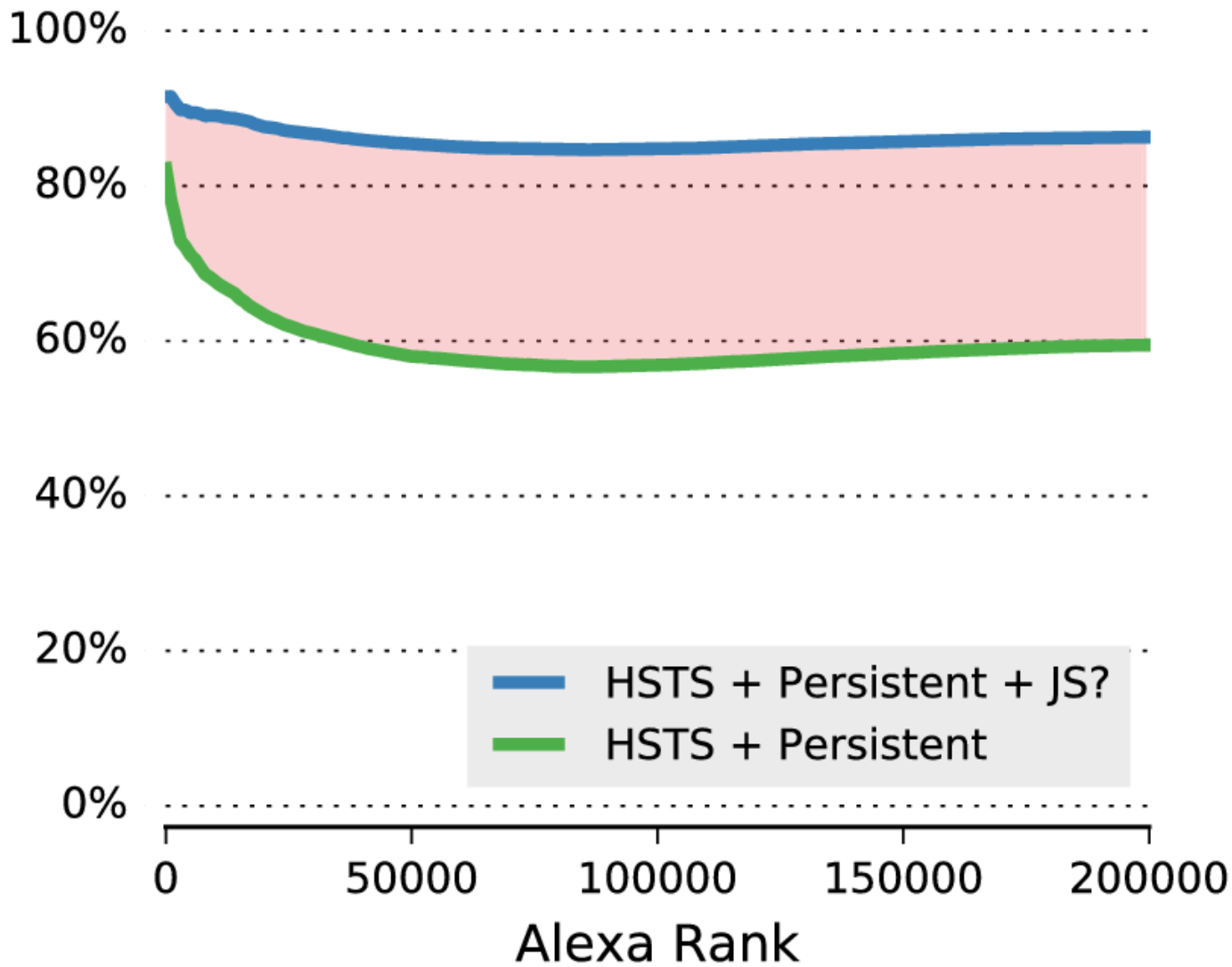
Captive Portal Detection

- A convenience feature
 - iOS since Version 4
 - Android since 4.2 – used default browser
 - Android since 5 – uses captive portal browser
- Test is easy to circumvent – HTTP!
 - http://clients3.google.com/generate_204
 - <http://captive.apple.com/hotspot-detect.html>

Which sites affected?

- Uses long-term (persistent) cookie
 - e.g., for session, tracking, or configuration
 - Can be set via
 - HTTP Header
 - Javascript
- Uses HSTS header

Sites vulnerable to history stealing

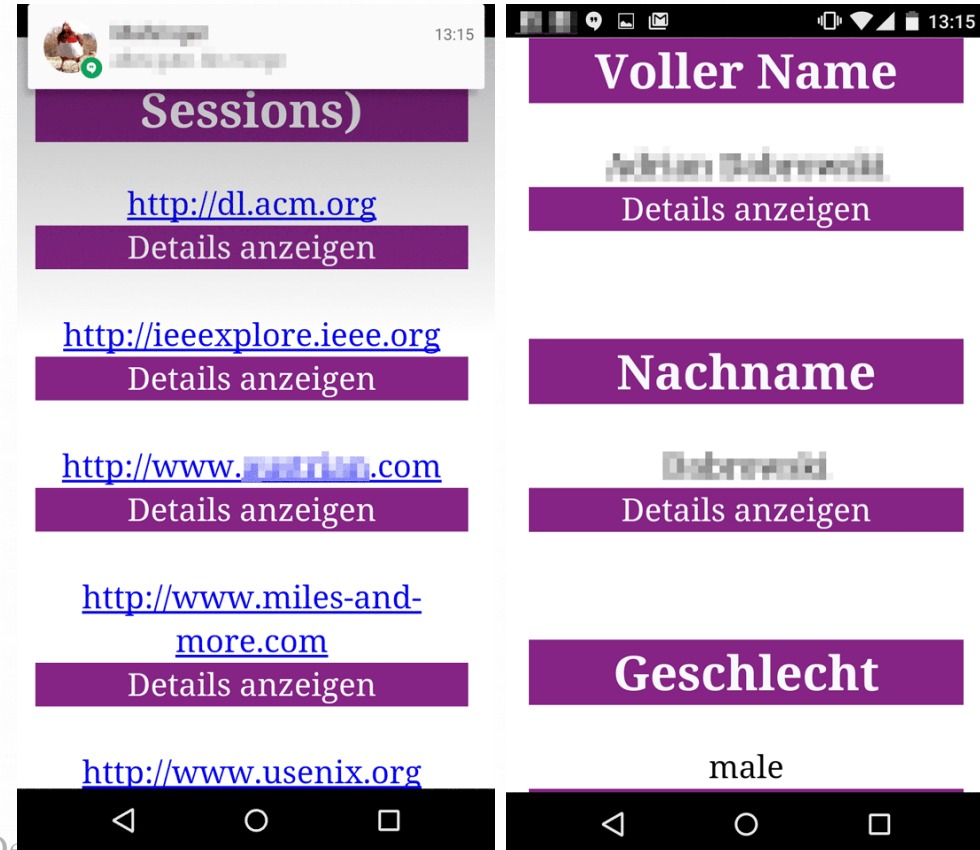


Top 1K
82-92%

Top 200K
59-86%

Implementation

- POC implementation
 - Speed up by...
 - Marking probe request with a special string
 - Returning 1x1 pixel or killing connections
 - Caching DNS
 - Still one DNS request for each probed site
- Added de-anonymization
 - e.g., via amazon.com



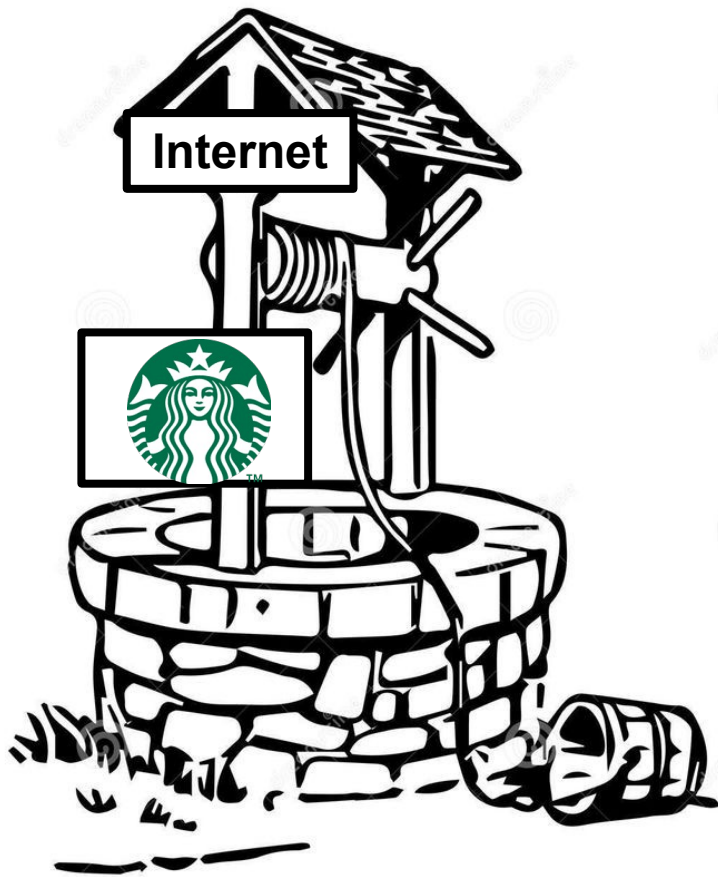
Solutions & Countermeasures?

Transmission

Hotspot Eco-System

Personal Protection

Hotspot Ecosystem



There is no standard for hotspot splash screen display

- Choose to present message in-band
- By redirecting/tampering with traffic
- Some do this also for SSL traffic
- e.g., via DNS `_portal.local`

Countermeasures

On the Client

- Better captive portal detection
 - Private browsing mode for portal
- Same-Site Cookies
 - Circumvention with one more fake indirection step
- Hotspot 2.0
 - Not widely supported
 - Solution for seamless roaming, not for showing banners and ads to the customer

On the Server

- HTTPS-only cookies
 - Google is changing an increasing number of Client-APIs to require HTTPS connections

Personal Countermeasures

- Do not use public WiFi!
 - Even when Encrypted!
If Password is public (e.g. same for every room in Hotel)
 - Privacy Mode for non-VPN connections
- 802.1x authentication the way to go
 - Only viable for private / campus / office / roaming wide installations
- Connect via mobile network
 - e.g., MiFi, or Tethering
 - Roaming packages, local SIM card

Conclusion

- Captive Portals (& MITM) can learn about the
 - current session
 - past browsing sessions
 - even for VPN users
- Side channels
 - Cookies
 - HSTS
- Alexa Top 1K
 - 82-92%
- Alexa Top 200K
 - 59-86%

HSTS and Cookie Side-Channels: Stealing Browser History

Adrian Dabrowski

Adrian.Dabrowski@tuwien.ac.at
adabrowski@sba-research.org

Twitter: @atrox_at