You are what you include:

Large-scale evaluation of remote JavaScript inclusions

Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, **Steven Van Acker**, Wouter Joosen, Christopher Kruegel, Frank Piessens, Giovanni Vigna






KATHOLIEKE UNIVERSITEIT LEUVEN DistriNet RESEARCH GROUP

Introduction: my USB stick

Introduction: browsers don't care

You are what you include:

Large-scale evaluation of remote JavaScript inclusions

Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, **Steven Van Acker**, Wouter Joosen, Christopher Kruegel, Frank Piessens, Giovanni Vigna






KATHOLIEKE UNIVERSITEIT LEUVEN DistriNet RESEARCH GROUP

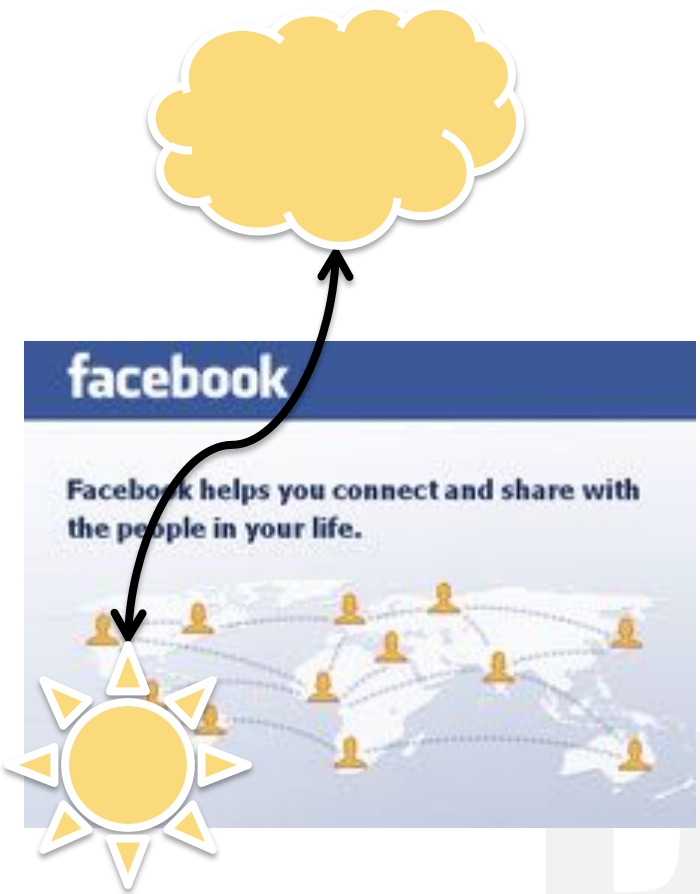
Outline

- JavaScript in a browser
  - … and motivation for an experiment
- Our experiment
- Our results
  - Some unsurprising results
  - Some weirdness
- Countermeasures

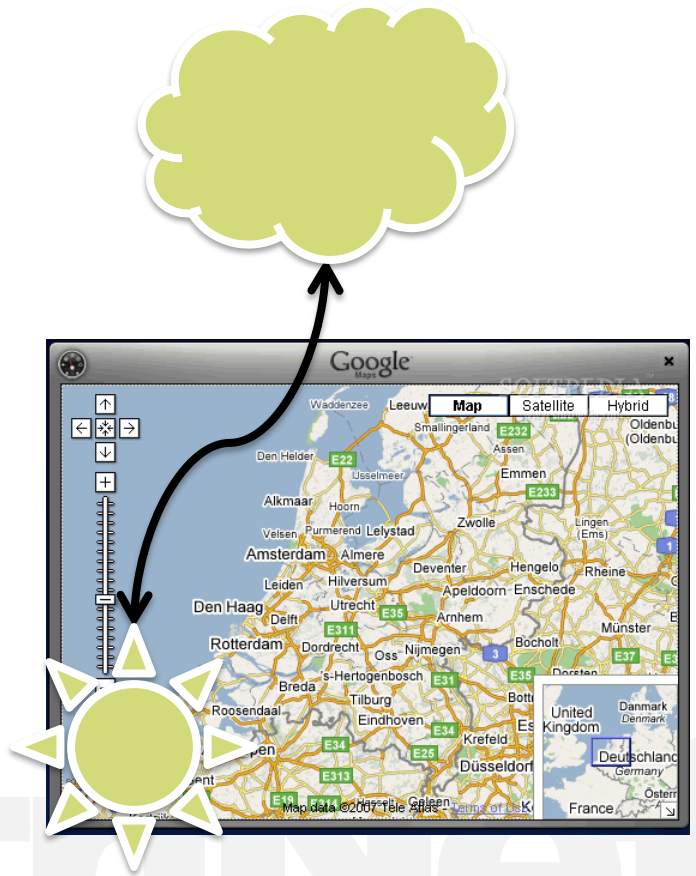
JavaScript in the browser

JavaScript in a browser: origins

Origin: http, facebook.com, 80



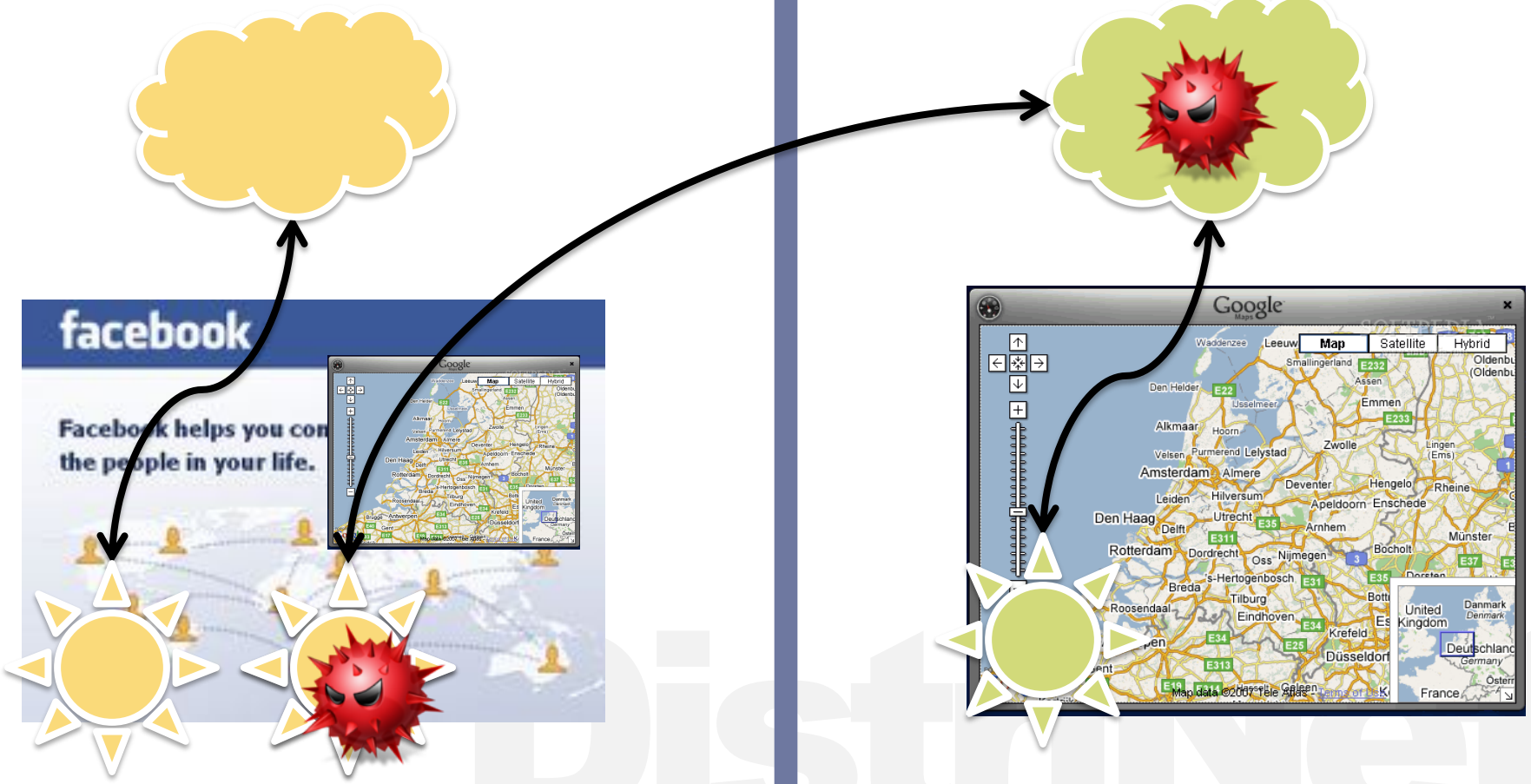
Origin: http, google-maps.com, 80



JavaScript in a browser: inclusions

Origin: http, facebook.com, 80

Origin: http, google-maps.com, 80



Motivation…

32 days…

Our experiment

Our experiment: questions

Given that remote JS inclusions happen…

… Should sites be trusting remote providers?

- Which third-party vendors do they currently trust?
- Are JS providers capable of securing their website? What is the quality of maintenance profile of each JS provider?
  - → Could a provider be attacked as a way of reaching a harder-to-get target?
- Are there attack vectors, in relation to remote inclusions, that we were not aware of ?
- How can one protect his web application?
  - → Are coarse-grained sandboxes sufficient?

Our experiment: crawler

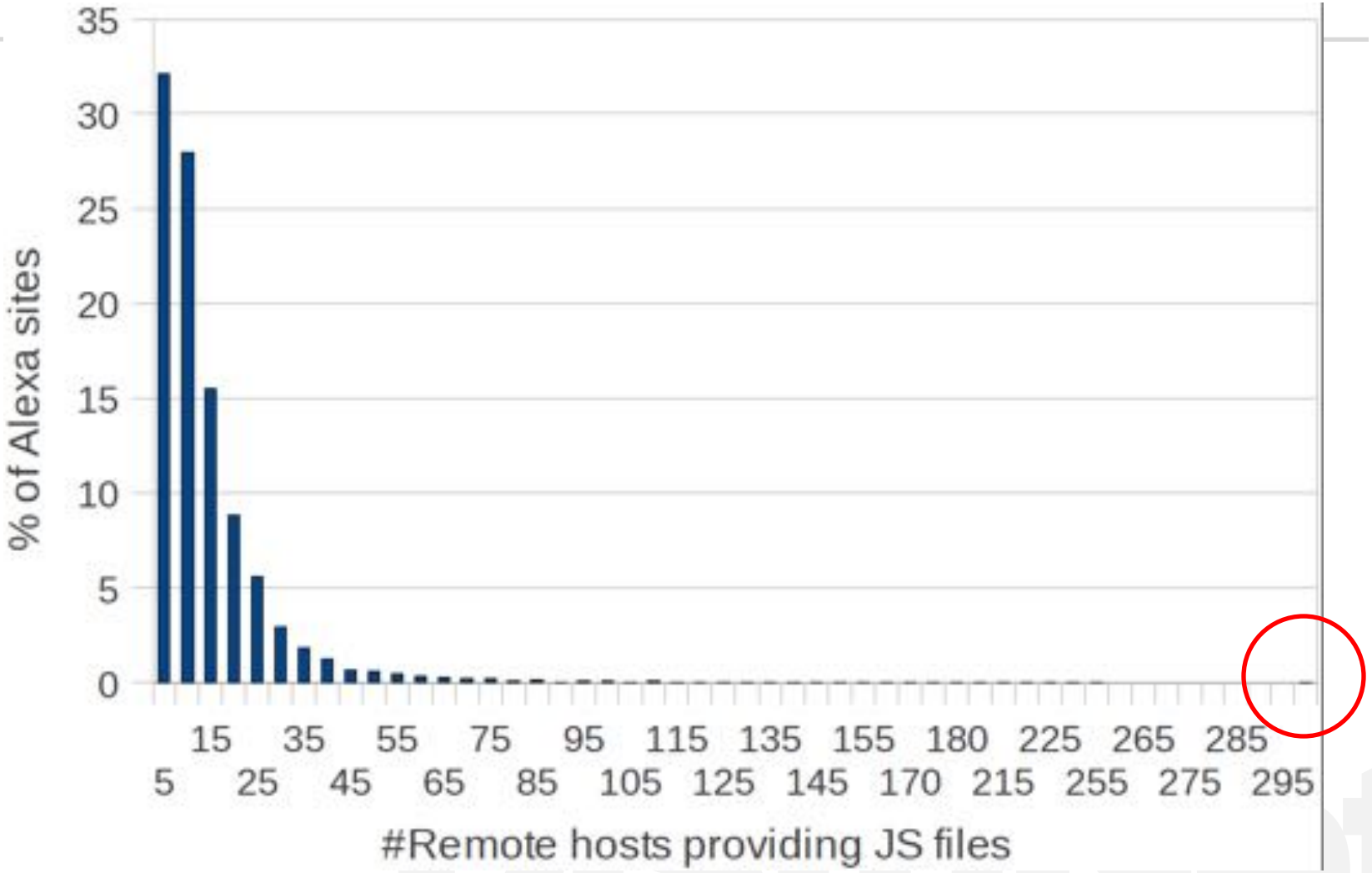
- Crawler requirements:
  - Download webpages
  - Log JavaScript inclusions
  - Execute JavaScript for dynamic inclusions
- HTMLUnit: JS-enabled headless browser in Java
- Queried Bing for max 500 pages of Alexa top 10000

Our experiment: some numbers

- Crawled over 3,300,000 pages belonging to the Alexa top 10,000
- Discovered:
  - 8,439,799 remote inclusions
  - 88.45% of Alexa top 10k uses at least 1 remote JS library
  - 301,968 unique JS files
  - 20,225 uniquely-addressed remote hosts

Results: unsurprisingly…

Results: how many remote hosts?



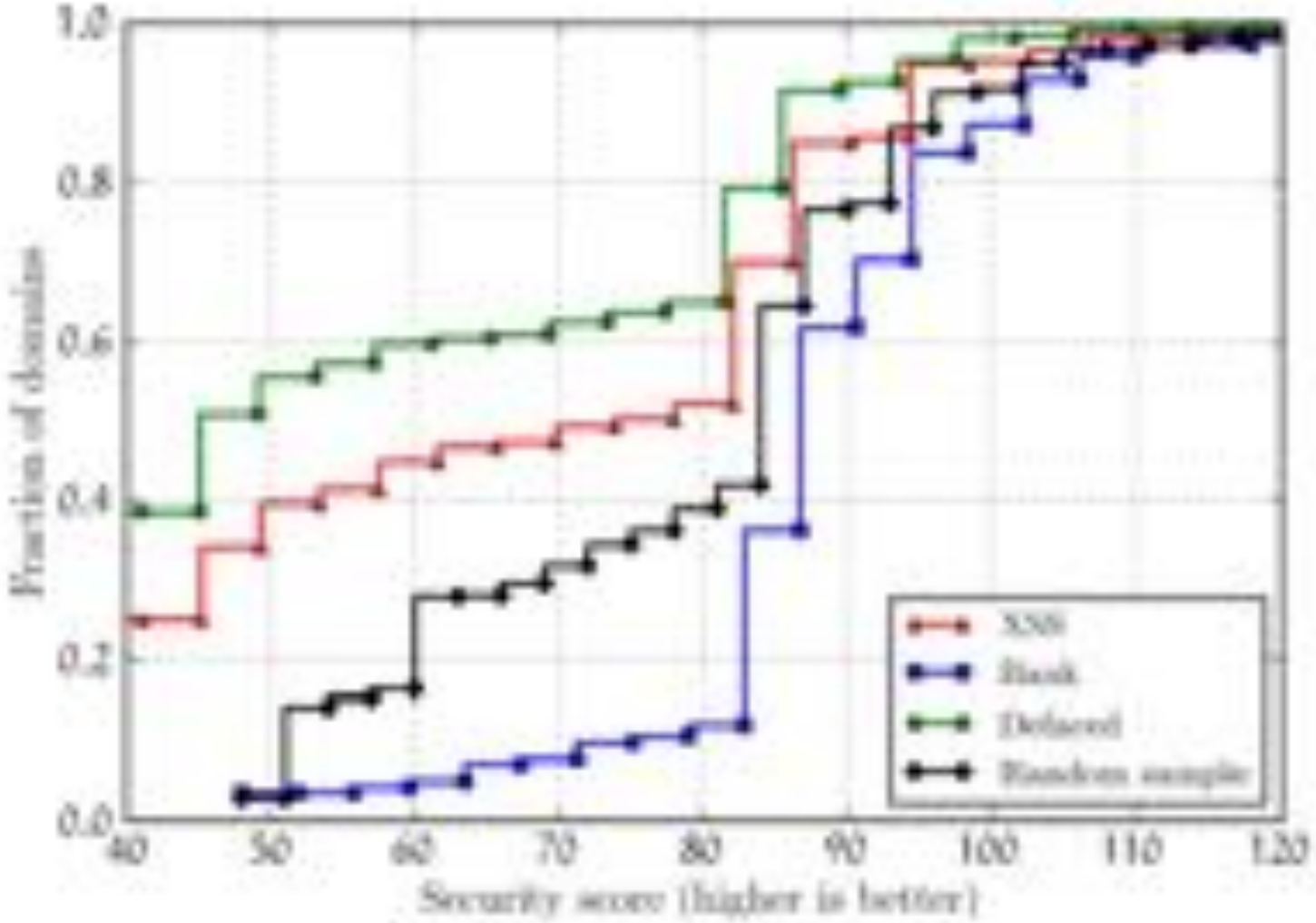
Results: Popular JavaScript includes

Offered service	JavaScript file	% Top Alexa
Web analytics	www.google-analytics.com/ga.js	68.37%
Dynamic Ads	pagead2.googlesyndication.com/pagead/show_ads.js	23.87%
Web analytics	www.google-analytics.com/urchin.js	17.32%
Social Networking	connect.facebook.net/en_us/all.js	16.82%
Social Networking	platform.twitter.com/widgets.js	13.87%
Social Networking & Web analytics	s7.addthis.com/js/250/addthis_widget.js	12.68%
Web analytics & Tracking	edge.quantserve.com/quant.js	11.98%
Market Research	b.scorecardresearch.com/beacon.js	10.45%
Google Helper Functions	www.google.com/jsapi	10.14%
Web analytics	ssl.google-analytics.com/ga.js	10.12%

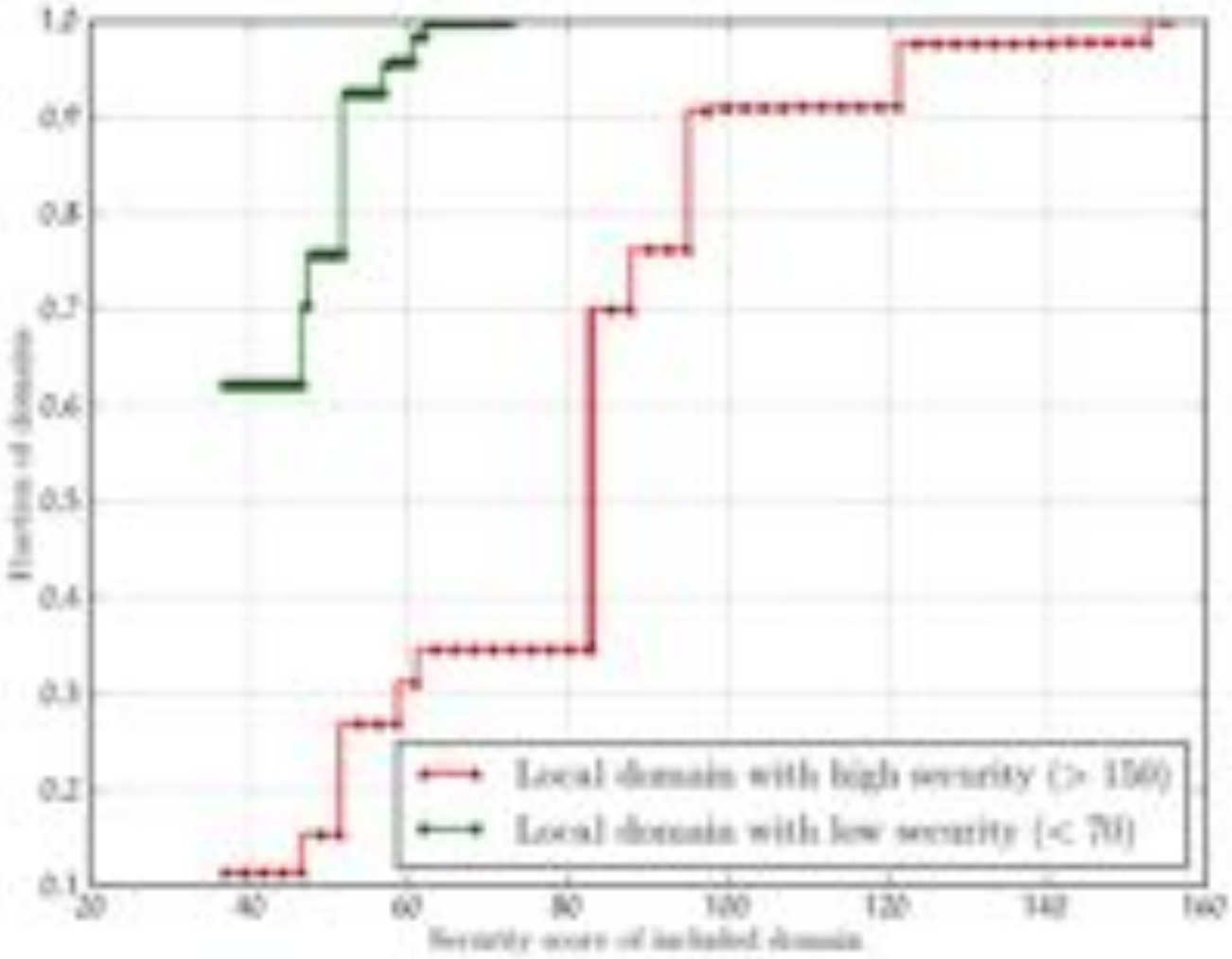
Results: quality of maintenance?

- Assumption: Unmaintained websites are easier to attack
- QoM indicator comprised of these factors:
  - Availability: DNS not expired, publicly-routable IP address
  - Cookies (at least one):
    - HttpOnly?
    - Secure?
    - Path & Expiration?
  - Anti-XSS & Anti-Clickjacking headers?
  - TLS/SSL implementation
    - Weak ciphers
    - Valid certificates
    - Strict Transport Protocol
  - Cache control when using TLS/SSL?
  - Outdated web servers?

Results: QoM in color!



Results: like attracts like



Results: weirdness!

Results: weirdness?

- In about 8.5 million records of remote inclusions, is there something that we didn't know?
- 4 Things! ☺
  - Cross-user & Cross-network Scripting
  - Stale domain-based inclusions
  - Stale IP-based inclusions
  - Typo-squatting Cross-Site Scripting

Weirdness: Cross-user Scripting

- <script src=http://localhost/script.js>
  - 133 records were found
  - 131 specified a port (localhost:12345), always greater than 1024
  - Attack:
    - Setup a web-server, listen to high ports, hack other users

Weirdness: Cross-network Scripting

- <script src=http://192.168.2.3/script.js>
  - 68 of them
  - Same as before, but now you just need to be in the same local network
- Who is doing that?
  - akamai.com
  - virginmobileusa.com
  - gc.ca (Government of Canada)

Weirdness: Stale IP-based remote inclusions

- What if the IP address of the host which you trust for JavaScript, changes?
  - The including page's scripts must also change
  - Do they?
- Manual analysis of the 299 pages
  - 39 addresses had:
    a) Not changed
    b) no longer provided JavaScript
       a) In 89.74%, we got a "Connection Timeout"

Weirdness: Stale domain-based inclusions

- What happens when you trust a remote site and the domain of that site expires?
  - Anyone can register it, and start serving malicious JS
  - Equal in power to the, almost extinct, stored XSS
    - Try proving in court that someone hacked you with that
- 56 domains found, used in 47 sites
  - 6 were identified as special cases (TXSS)

Scared yet?

Weirdness: Typo-squatting XSS (TXSS)

- Unfortunately… developers are humans
  - <script src=http://googlesyndicatio.com/...>
- Typo-squatting
  - registering domains that are mistypes of popular domains
  - Serve ads, phishing, drive-by downloads etc. to users that mistype the domain

Weirdness: TXSS examples found…

Intended domain	Actual domain
googlesyndication.com	googlesyndicatio.com
purdue.edu	purude.edu
worldofwarcraft.com	worldofwaircraft.com
lesechos.fr	lessechos.fr
onegrp.com	onegrp.nl

	Googlesyndicatio.com (15 days)
Unique visitors	163,188
Including domains	1185
Including pages	21,830

Countermeasures

Countermeasures

- Problems with remote inclusions
  - Never the visitor's fault
  - A developer can mess up
    - Cross-user, cross-network and TXSS
  - The remote host can mess up
    - Low security, expiration of domain names
- How to protect one's self?
  i. Sandbox remote scripts
  ii. Download them locally

Countermeasures: sandboxing

- Is it feasible?
- What are the current requirements of legitimate scripts?
- Study the top 100
  - → Automatically study each script
    - JavaScript wrappers + stack trace
  - → Find out what sensitive resources they access
    - Cookies, Storage, Geolocation, Eval, document.write
  - → Is containment possible?

… sandboxing: Access to resources

JS Action	# of Top scripts
Reading Cookies	41
`document.write()`	36
Writing Cookies	30
`eval()`	28
`XHR`	14
Accessing LocalStorage	3
Accessing SessionStorage	0
Geolocation	0

Coarse-grained sandboxing is useless here, legitimate scripts and attackers act the same way ☹

Countermeasures: local copies

- Study the frequency of script modifications
  - → Discover overhead for administrator
- Top 1,000 most-included scripts (803)
  - → Download every script three consecutive times and remove the ones that changed all three times
  - → Study the rest for a week
- 10.21% were modified
  - → 6.97% were modified once
  - → 1.86% were modified twice
  - → 1.83% were modified three or more

89.79% was never modified!
96.76% at most once

Conclusions

Conclusions

- Remote inclusions mean, almost unconditional, trust
  - Think twice before including something from a remote host

- Do **NOT**:
  - Include from 127.0.0.1 or private networks
  - Include from IP addresses
  - Include from stale domains
  - Include from typodomains
  - Include from questionable JS providers

- Do:
  - Make local copies
  - Sandbox 3rd party JS if it is feasible
  - Have hope: sleep sound tonight

Thank you!

Questions?