Large Scale Analysis of CORS Misconfigurations

Jens Müller

hgi NDS

http://www.nds.rub.de/

Motivation

- HTTP security headers
  - **`X-Frame-Options`**
  - **`X-Content-Type-Options`**
  - **`X-XSS-Protection`**
  - **`Referrer-Policy`**
  - **`CSP, HSTS, HPKP`**
  - …

IF THERE WAS A HTTP HEADER TO REMOVE ALL SECURITY
Yup.
WOULD TOP SITES USE IT?

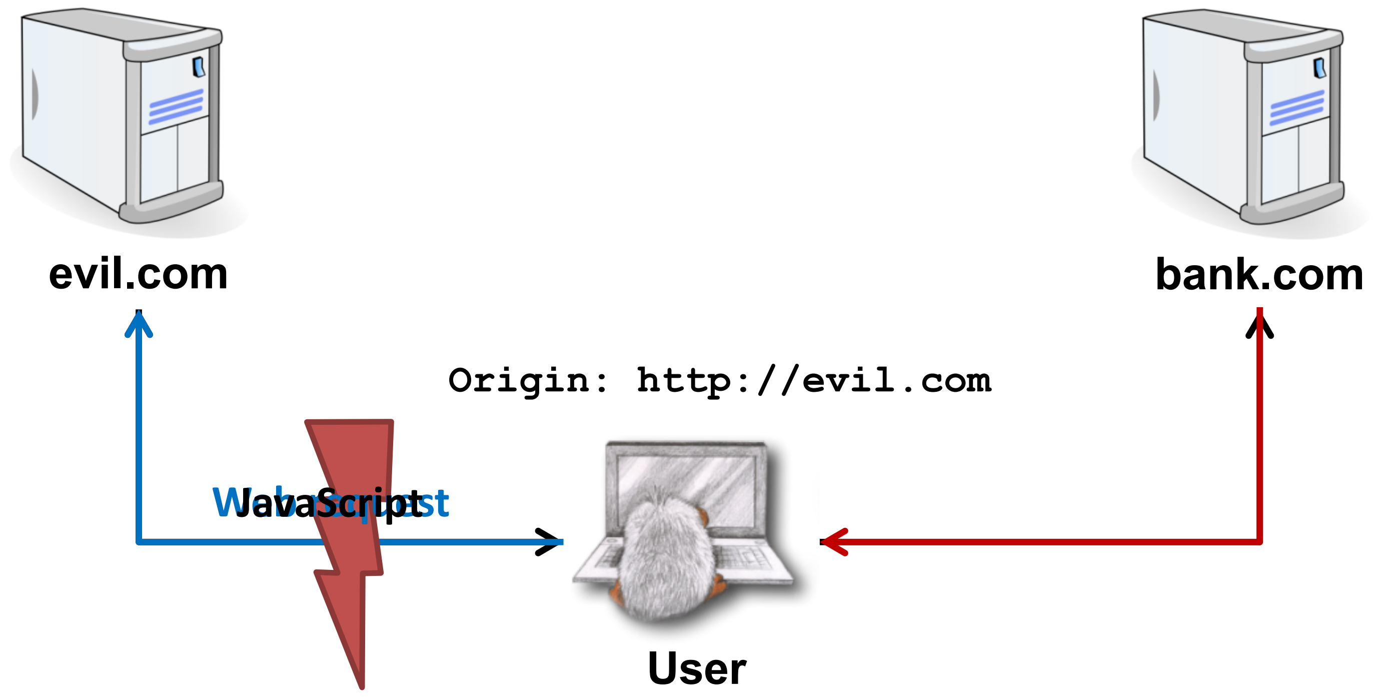
Overview

What is CORS?

- Cross-Origin Resource Sharing
- Enables web servers to explicitly allow cross-site access to a certain resource
- Punches holes into Same-Origin Policy

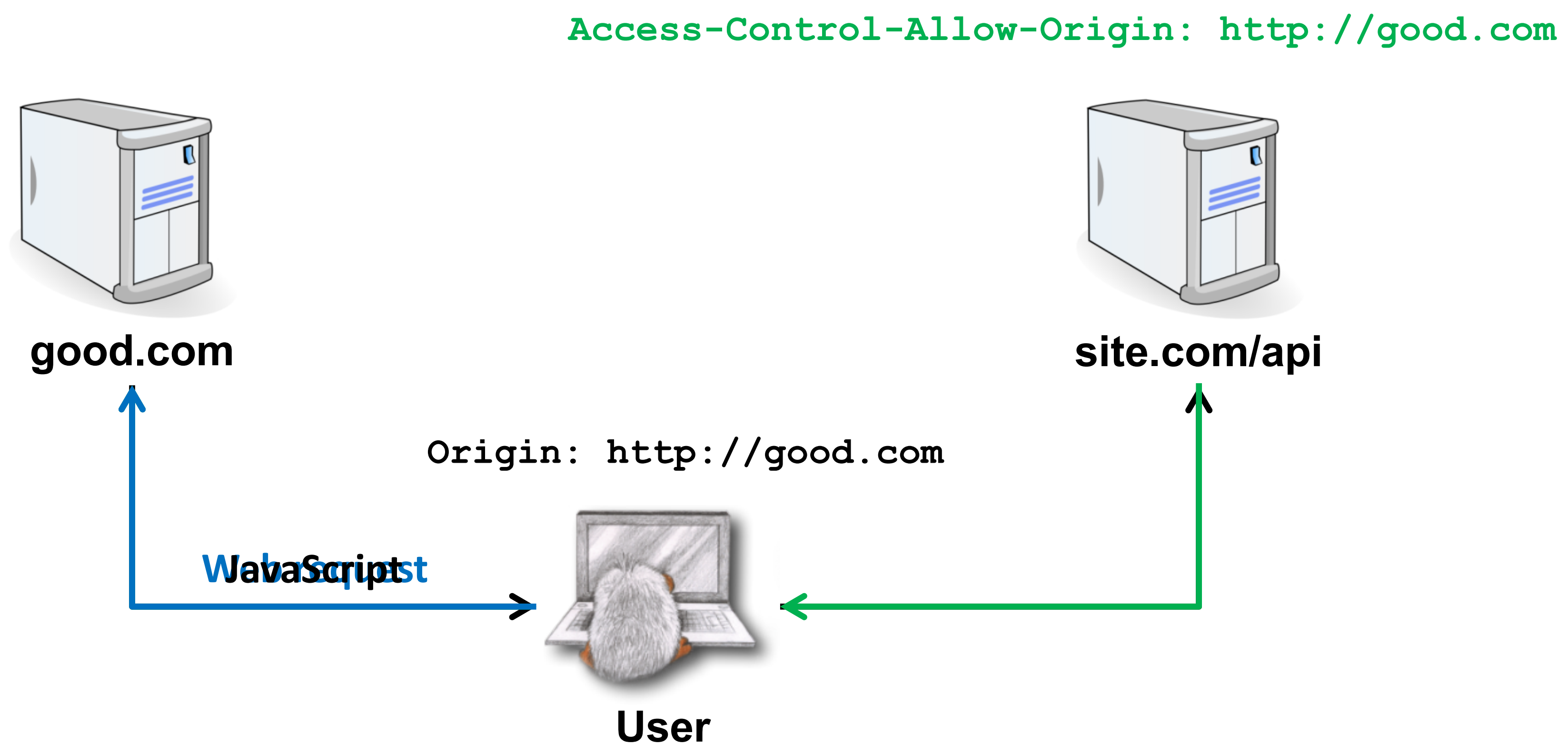
Example

- **Same Origin Policy:** Scripts can only access data from the same origin (protocol, domain, port)



Cross-Origin Resource Sharing

- CORS-based web API access



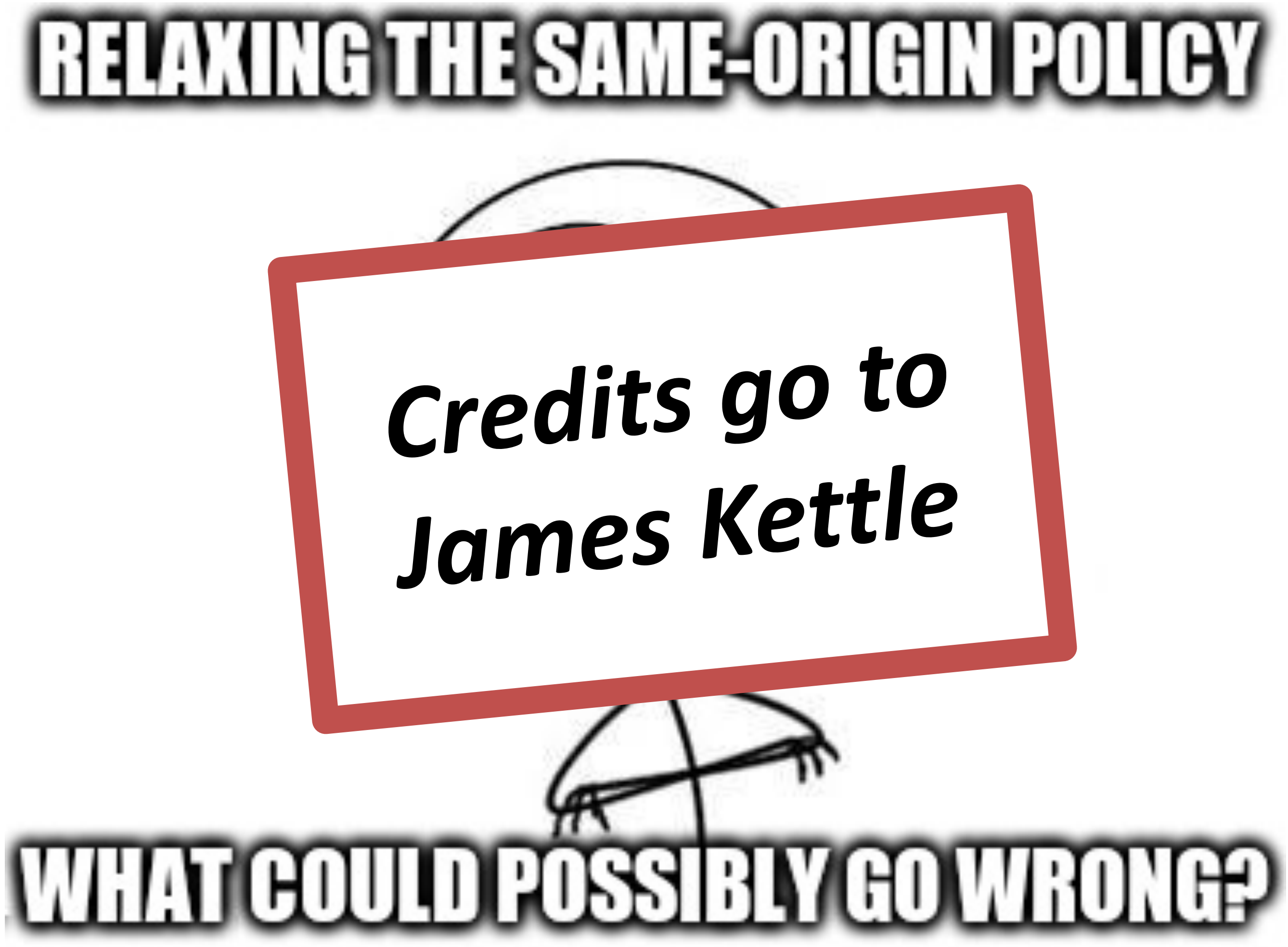
CORS HTTP headers

- **`Access-Control-Allow-Origin (ACAO)`**
  - Which URI is allowed access?
- **`Access-Control-Allow-Credentials (ACAC)`**
  - Access with (session) cookies?
- Some more **`Access-Control-`**… headers

Overview

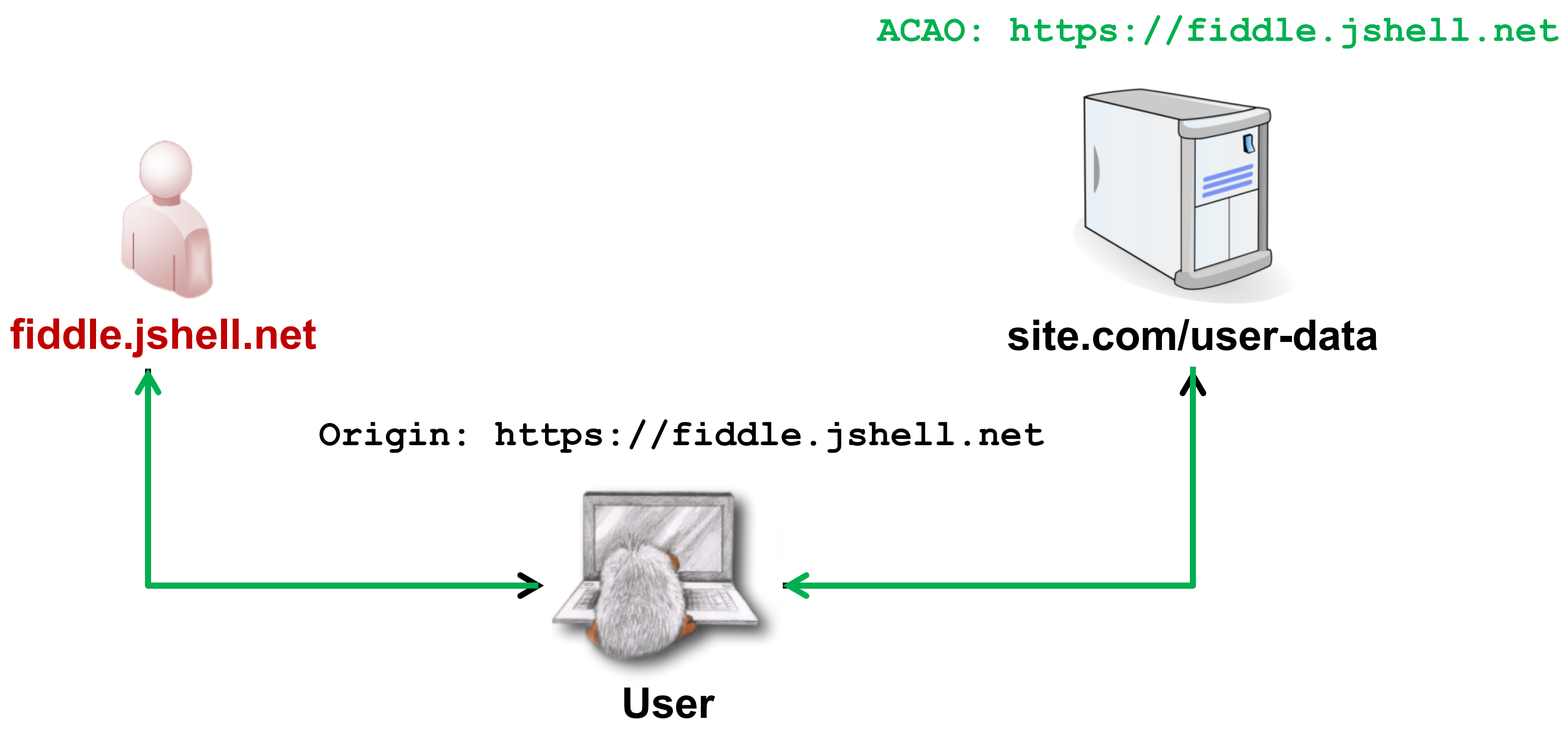
1. Background
2. Misconfigurations
3. CORStest
4. Evaluation
5. Conclusions

Misconfigurations


RELAXING THE SAME-ORIGIN POLICY
Credits go to James Kettle
WHAT COULD POSSIBLY GO WRONG?

Developer backdoor

- Insecure developer/debug origins allowed


ACAO: https://fiddle.jshell.net
fiddle.jshell.net
site.com/user-data
Origin: https://fiddle.jshell.net
User

Allowing access to multiple sites

- Allow all origins
  - `ACAO: *`
  - but never with credentials (therefore mostly harmless)
- Invalid configurations:
  - `ACAO: site1, site2`
  - `ACAO: *.site`
- Solution:
  Dynamically return `ACAO` based on `Origin`

Subdomains allowed

- `sub.domain.com` allowed access
  - exploitable if XSS in *any* subdomain

Post/pre domain wildcard

- `notdomain.com` is allowed access
  - can simply be registered by the attacker
- `domain.com.evil.com` is allowed access
  - can be simply be set up by the attacker

Origin reflection

- The origin is simply echoed in **`ACAO`** header
  - any site is allowed to access the resource

Null misconfiguration

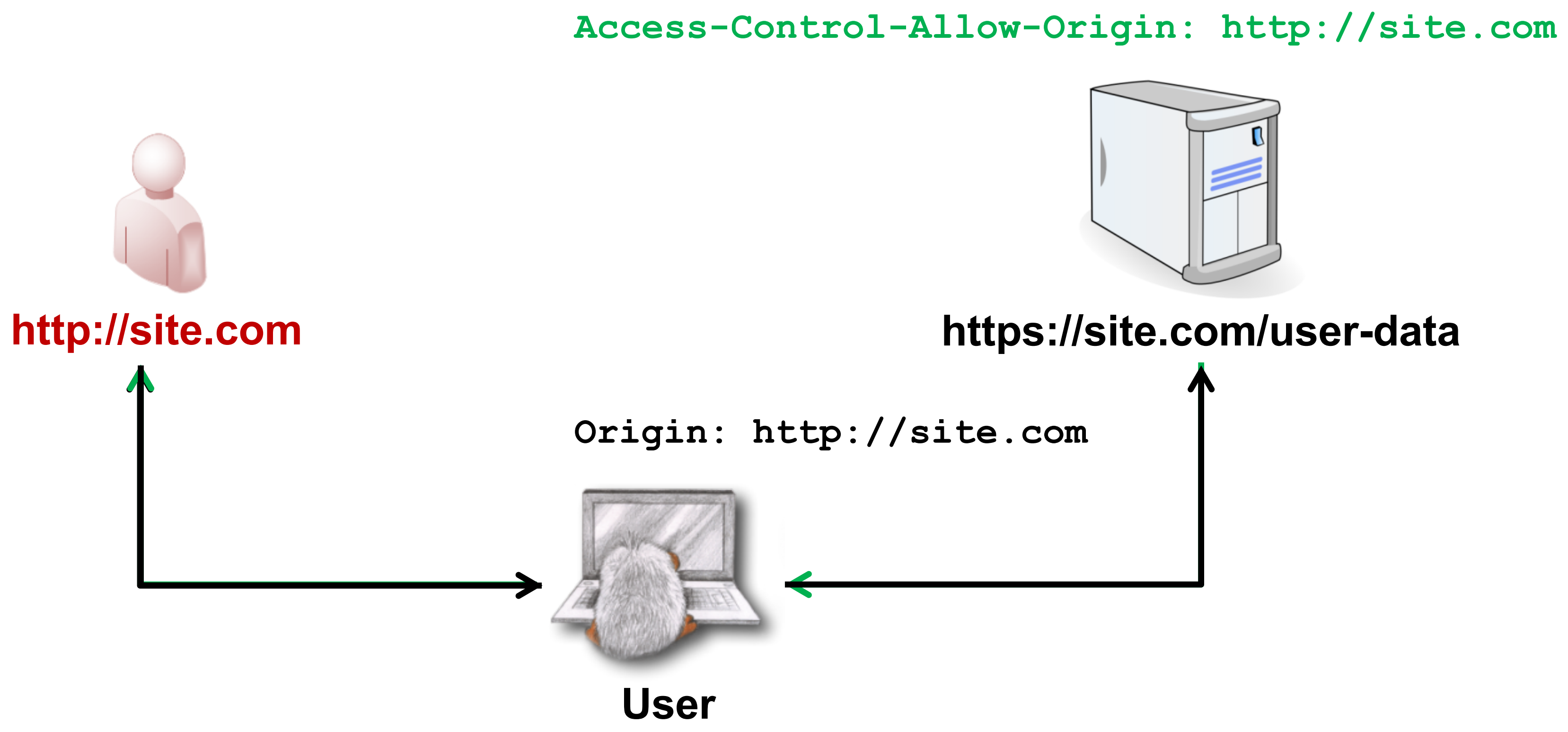
- `ACAO: null` to allow local HTML files
- `null` origin can be forced using an iframe
  - any site is allowed to access the resource
- `null` may be returned by software (Node.js)

Protocol-relative URLs

- `ACAO: //` returned by some websites
- How should browsers deal with this?
  - IE, Edge: deny all origins
  - FF, Ch, Sa, Op: allow all

Non-ssl sites allowed

- A **`http`** origin is allowed access to a **`https`** resource, allows MitM to break encryption



Access-Control-Allow-Origin: http://site.com
http://site.com
https://site.com/user-data
Origin: http://site.com
User

Overview

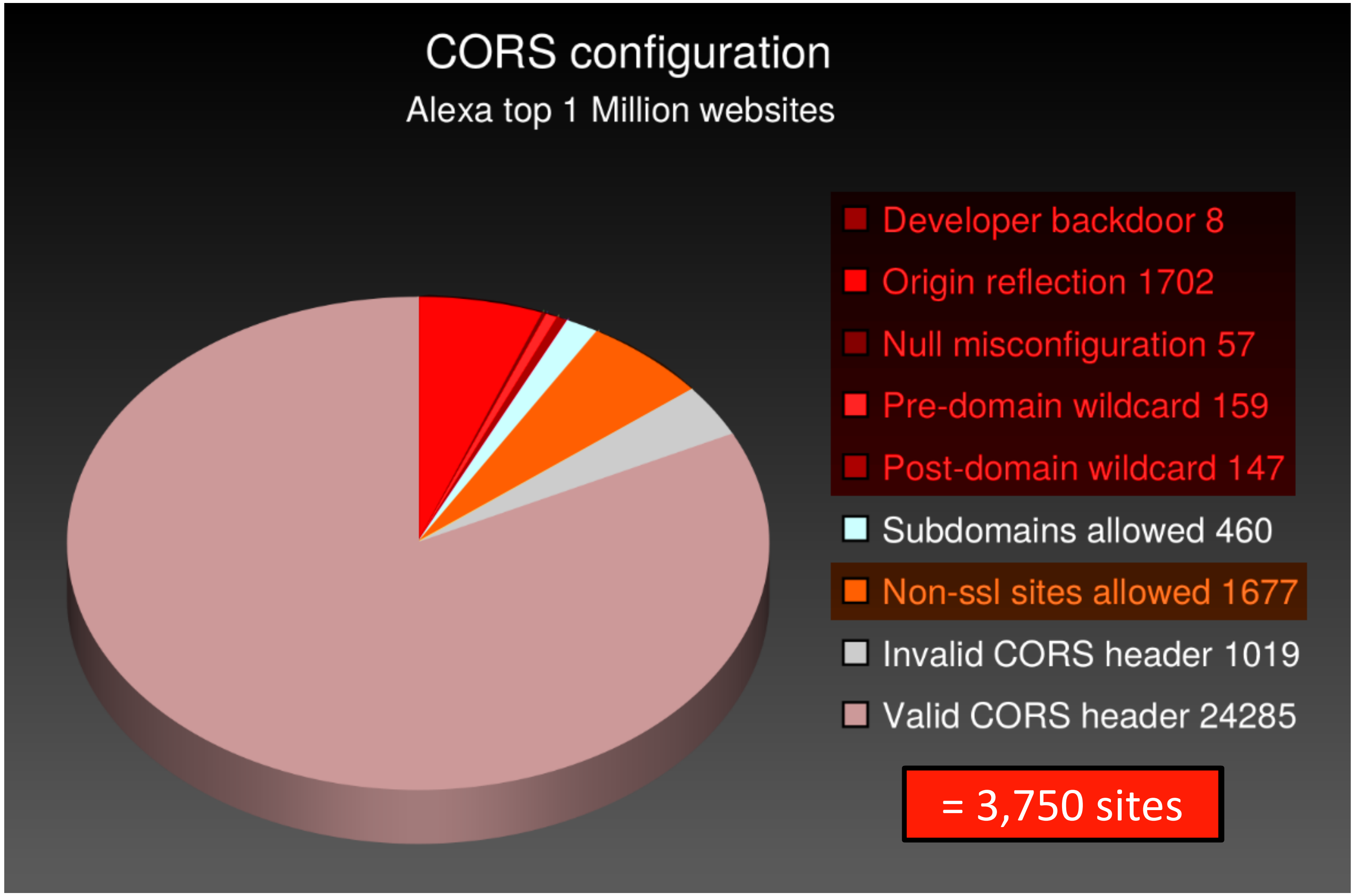
CORStest

- Simple CORS misconfiguration scanner
- https://github.com/RUB-NDS/CORStest
- Sends requests with various **`Origins`** checks for the **`ACAO/ACAC`** responses

Demo time

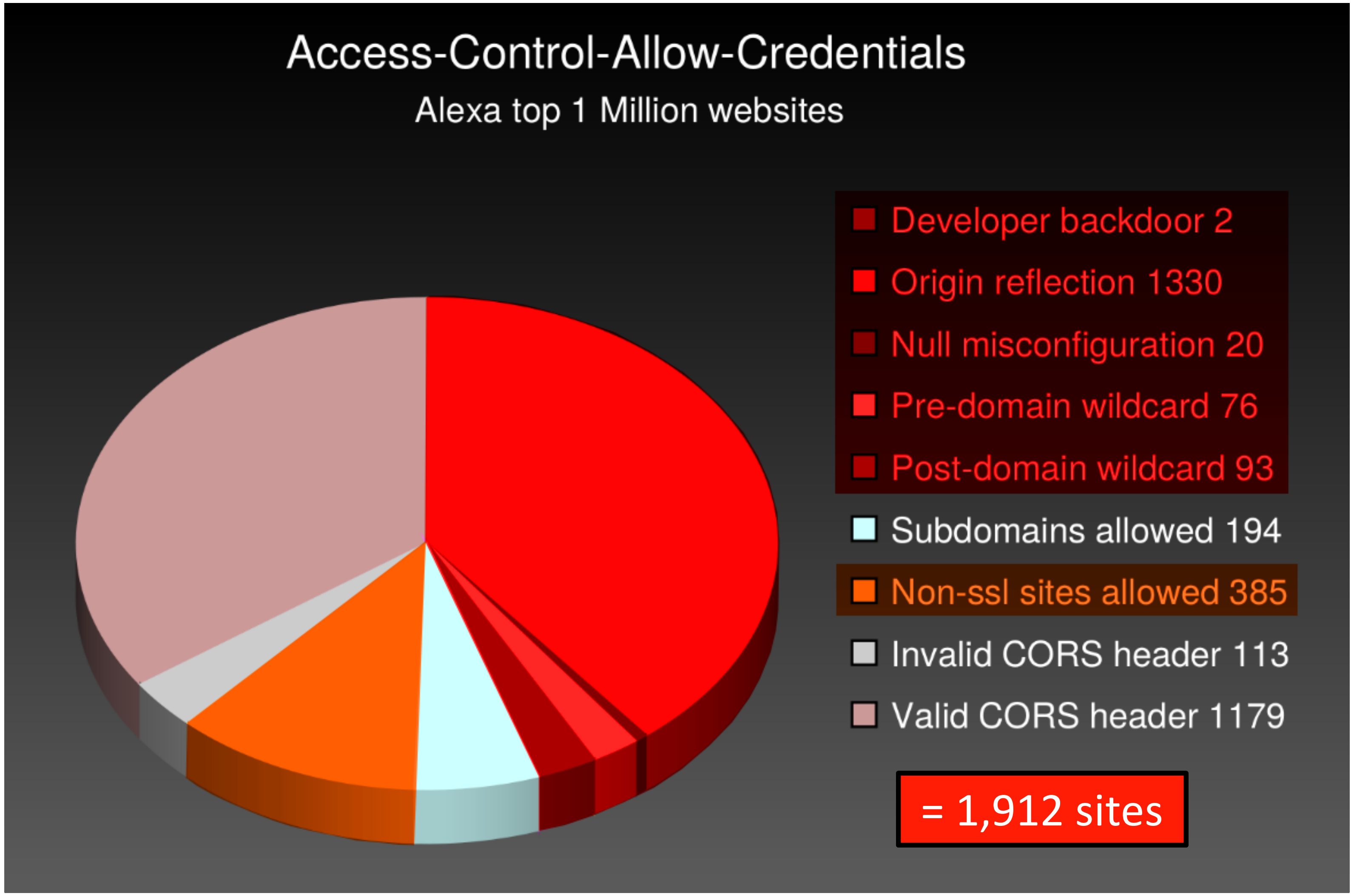
Overview

Evaluation: Alexa top 1m websites



= 3,750 sites

Evaluation: Alexa top 1m with credentials



Popular vulnerable sites

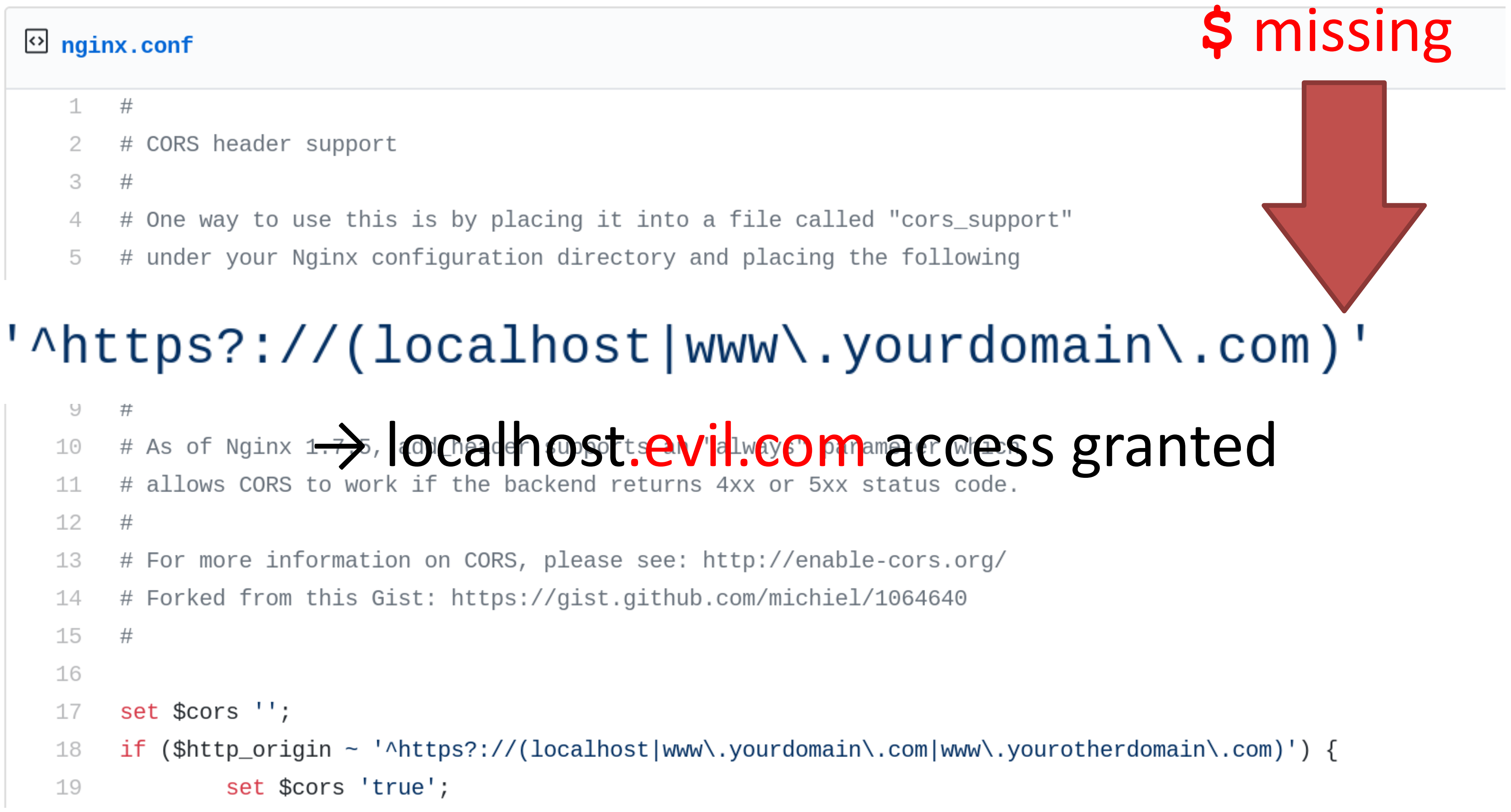
cheaptickets.de
ail.bg
abendzeitung-muenchen.de
walmart.com
provinzial.de
oard.com
loanframe.com
profile.accounts.firefox.com
m
login.worldpay.com
netbank.de
helpling.de
bitssa.com
onestarnationalbank.com
udacity.com
esa.io
moneyversed
apttus.c
staffhub.co
fedex.com
9cloud.us
kaspay.com
coinplug.com
orpay.co
btcclick
bungie.ne
zuto.com
eltarif.de
ypax.info
wallet.baidu.com
k.com
ego.com
booking.com
rg
payoffshore.com
coinalarr
ctf365.com
garnier.de
assenversicherung.de
obamaca
agoda.com
re.paycor.com
.microsoft.com
uberall.com
nt.rapid7.c
yummly.com
citypay.com
nspire.c
ankofireland.c
pixieset.com
transform.microsoft.com
paytop.com
aeftskunden.te
flat.io
.de
porsche.com
academia.edu
om
fantrax.com
dzpay.
transferwise.c
planted.c
native-instruments.de
k.com
cloud.net
fundly.com
om
crystalgraphics.com
alipay.com
events.att.co
token
slice.com
-dresden.de
officerreports.net
wetransfer.com
schwarzwaelder-bote.de

Reporting on a medium scale

- Had to notify ~~1,912~~ 1,500 websites
- How to do this? Contact manually?
  - **`security@`**, **`support@`**, **`info@`**, **`privacy@`**
- About 300 websites fixed the flaw…
- Some did not want to believe:
  - Kevin has resolved your ticket: *“We are fully PCI-DSS compliant and have passed all scans”*
  - *“We use the most secured cloud servers and military grade encryption to backup your data”*

Causes for CORS misconfigurations

Example Nginx configuration for adding cross-origin resource sharing (CORS) support to reverse proxied APIs



nginx.conf

$ missing

```
#
# CORS header support
#
# One way to use this is by placing it into a file called "cors_support"
# under your Nginx configuration directory and placing the following
```

```
'^https?://(localhost|www\.yourdomain\.com)'
```

```
#
# As of Nginx 1.7.5, add_header supports an "always" parameter which
# allows CORS to work if the backend returns 4xx or 5xx status code.
#
# For more information on CORS, please see: http://enable-cors.org/
# Forked from this Gist: https://gist.github.com/michiel/1064640
#

set $cors '';
if ($http_origin ~ '^https?://(localhost|www\.yourdomain\.com|www\.yourotherdomain\.com)') {
        set $cors 'true';
```

→ localhost.evil.com access granted

Causes for CORS misconfigurations

- CORS in Action contains examples such as **`var originWhitelist = ['null', …]`**
- Rack::Cors maps **`origins ''`** or **`origins '*'`** into reflecting all origins (+ CVE-2017-11173)
- crVCL PHP Framework just checks if allowed origin string is contained in **`Origin`** value

Invalid headers

- Invalid (creative) **`ACAO`** values we observed:
  - **`self, true, false, undefined, None, 0, (null), domain, origin, SAMEORIGIN`**

Overview

Conclusions

- There is a lot of confusion on CORS
- It’s too easy to misconfigure CORS
- Can remove all your web security
- **`ACAO: *`** is mostly harmless

Thanks for your attention...

CORStest

- https://github.com/RUB-NDS/CORStest

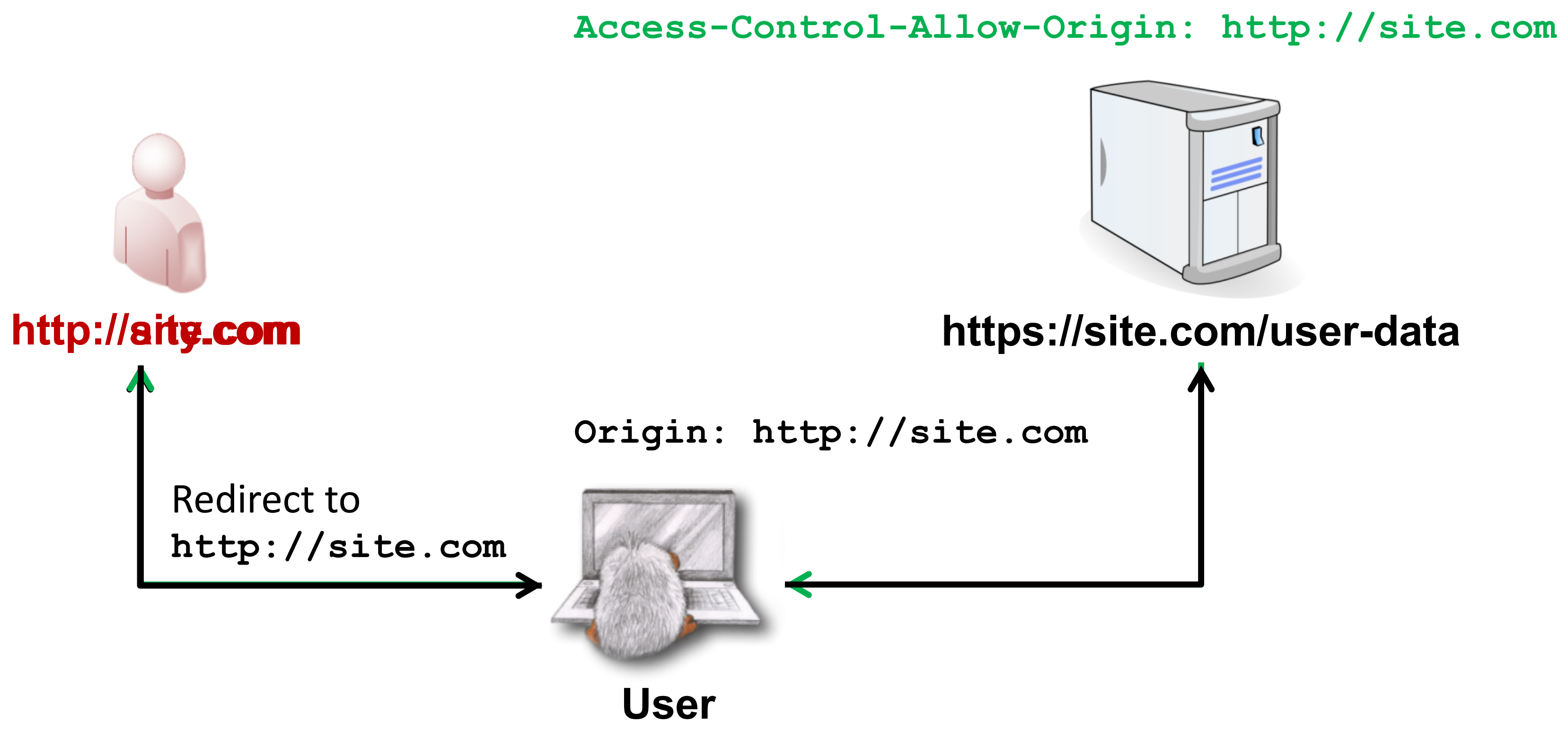
Questions?

Some popular sites

- Online banking, insurance, bitcoins, payment and US state's tax filing sites vulnerable:
  - sparkassenversicherung.de, bitcoinpay.com, coinplug.com , bankofireland.com, korpay.com, lonestarnationalbank.com, moneymonk.nl, netbank.de, paytop.com, transferwise.com, citypay.com, payoffshore.com, nystax.gov, id.net, booking.com, microsoft.com, yandex.com, geschaeftskunden.telekom.de, agoda.com, fedex.com, adidas.de, dasoertliche.de, …

Non-ssl sites allowed

- A **`http`** origin is allowed access to a **`https`** resource, allows MitM to break encryption


`Access-Control-Allow-Origin: http://site.com`
http://aity.com
https://site.com/user-data
`Origin: http://site.com`
Redirect to
`http://site.com`
User