



OWASP

Open Web Application
Security Project

If You Can't Beat 'Em Join 'Em:

Practical Tips for Running a Successful Bug Bounty Program

Grant McCracken and Daniel Trauner

AppSecUSA 2016

Grant

- Solutions Architect @Bugcrowd
 - Formerly an AppSec Engineer
- Before that, WhiteHat
- Traveling, music, stuff.



Dan

- Senior AppSec Engineer @Bugcrowd
- Before that, Software Security @HP
 - Static analysis – lots of languages
 - Focused on Apple iOS
- Art history and collecting

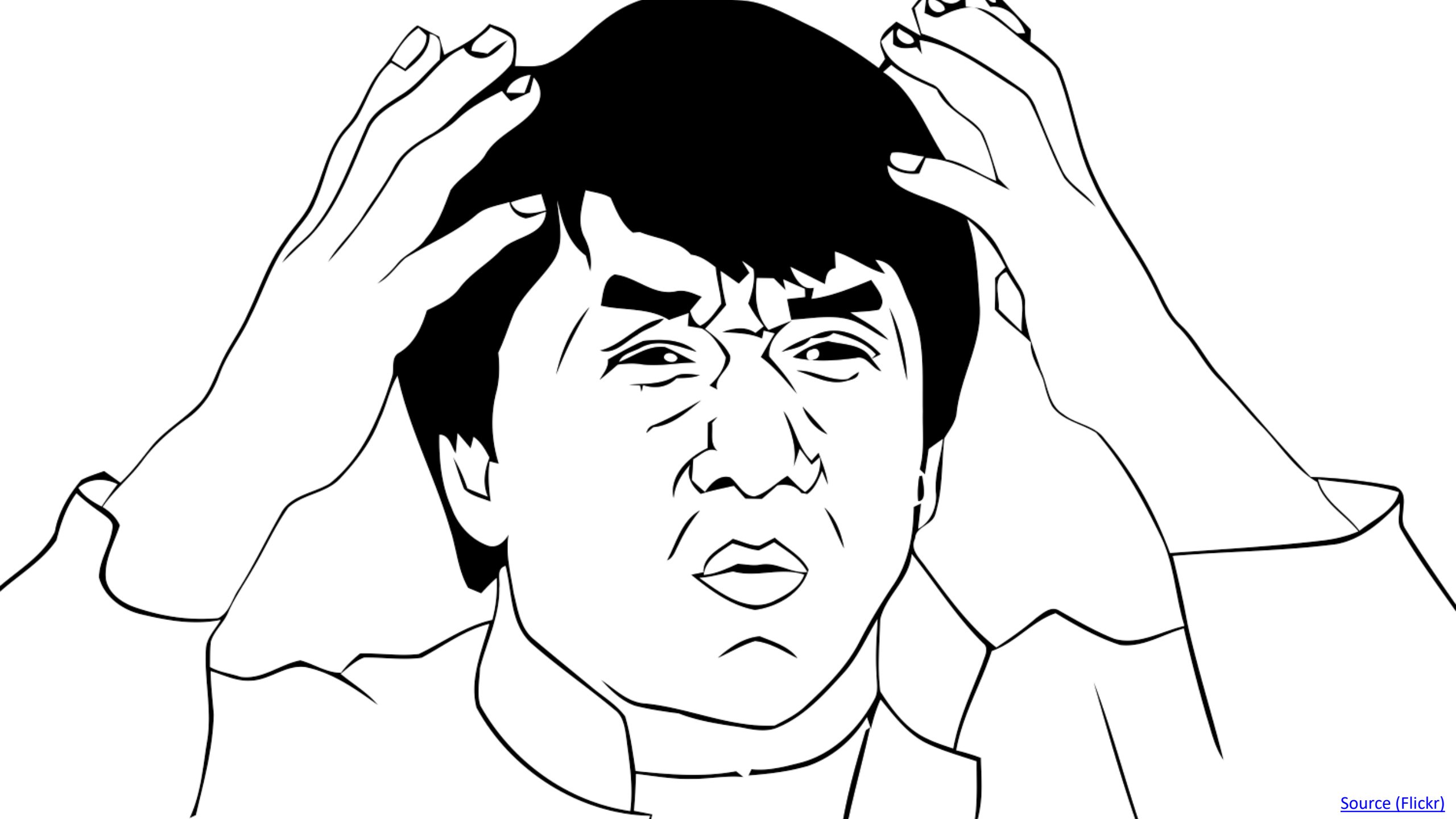


CONNECT

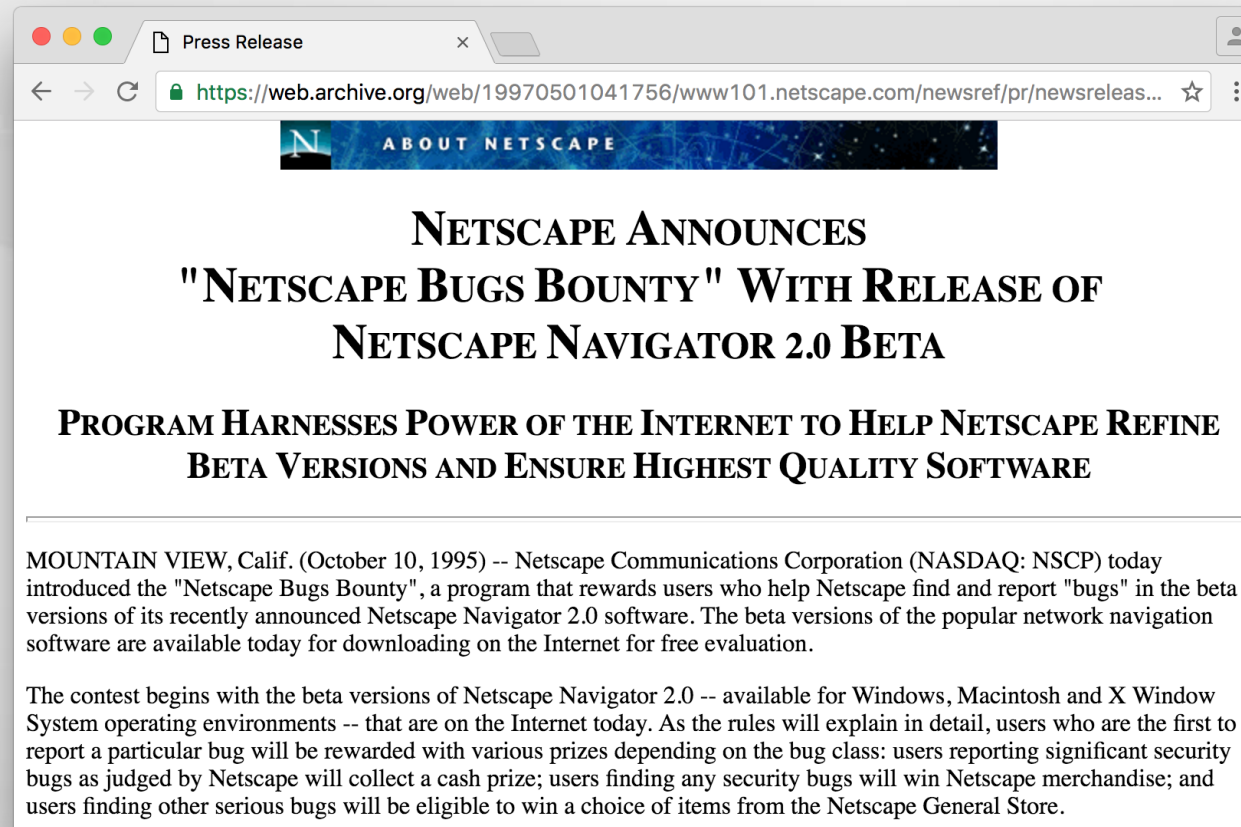
LEARN

GROW

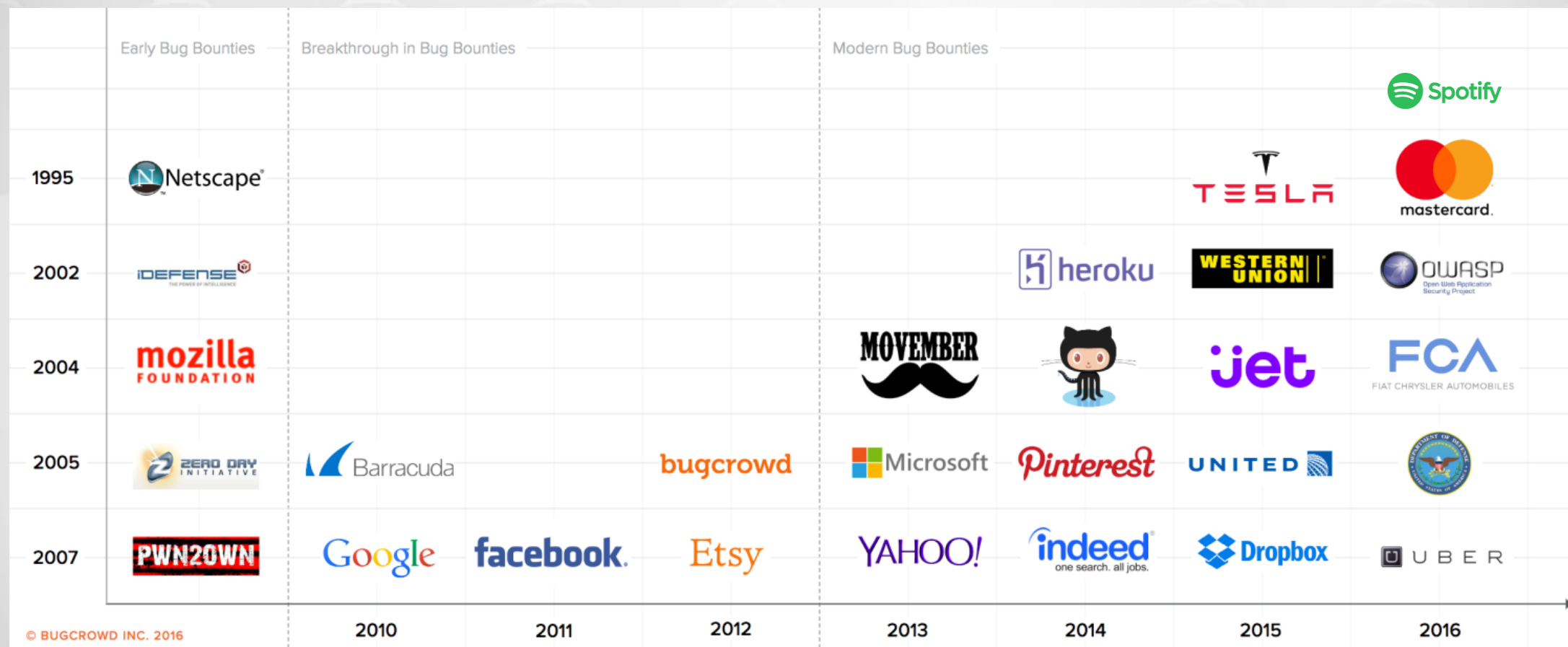
BUG BOUNTY PROGRAMS



Netscape "Bugs Bounty"



An (Abbreviated) History of Bug Bounties Since 1995



CONNECT

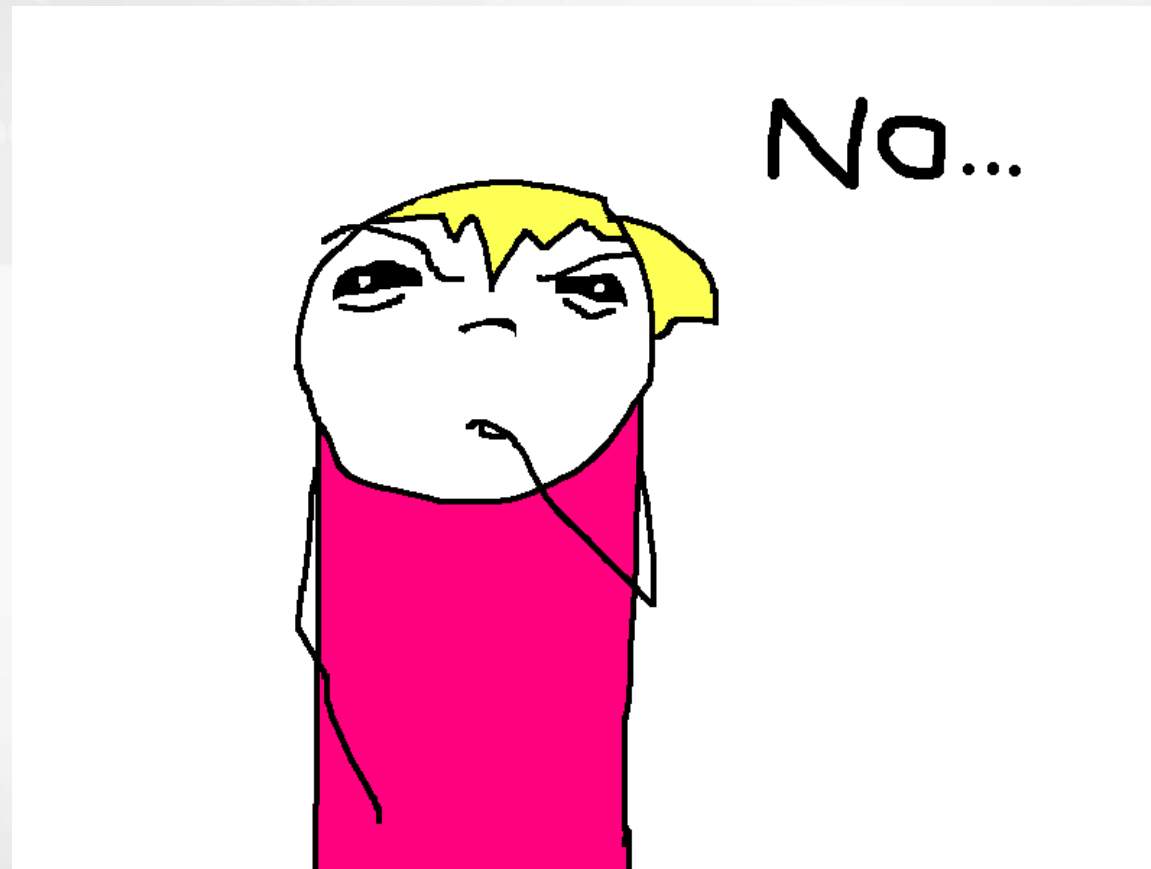
LEARN

GROW

WHY?



Do you really want to let people attack you?



[Source \(Hyperbole and a Half\)](#)

Yes! (They're doing it anyways...)



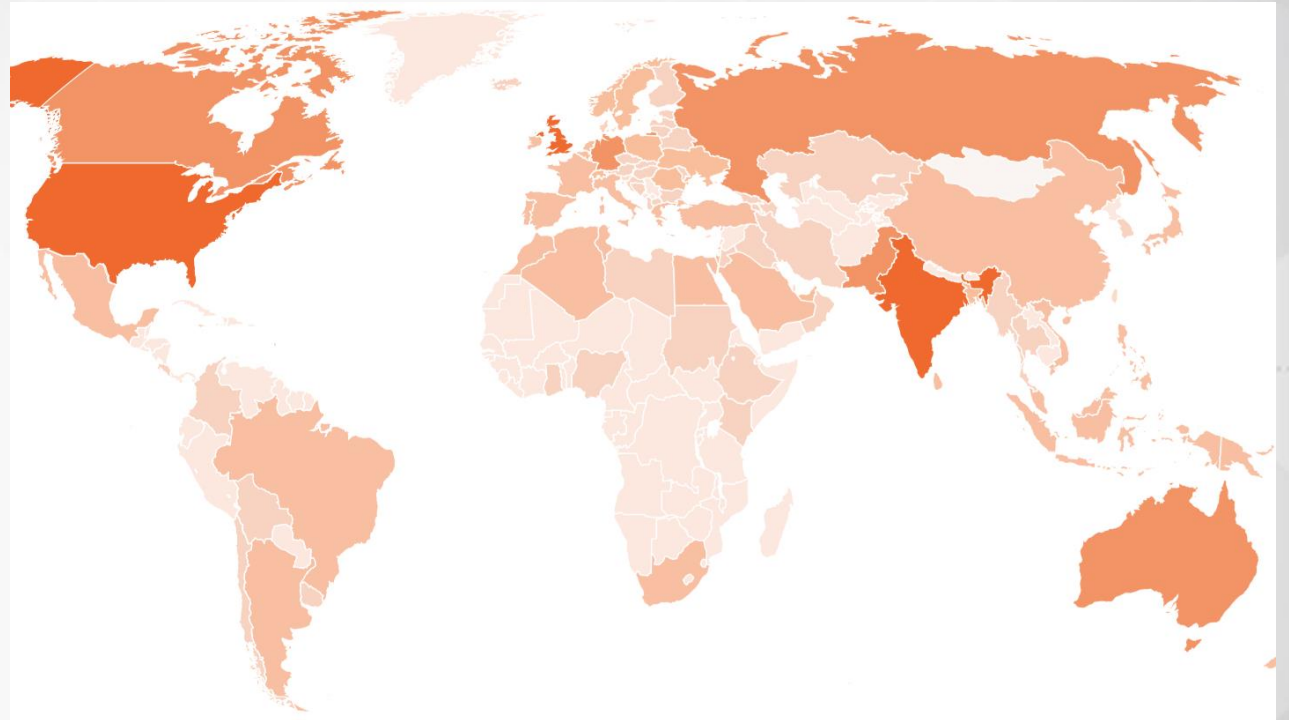
[Source \(Hyperbole and a Half\)](#)

You ~~vs.~~ and Them



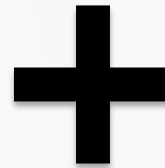
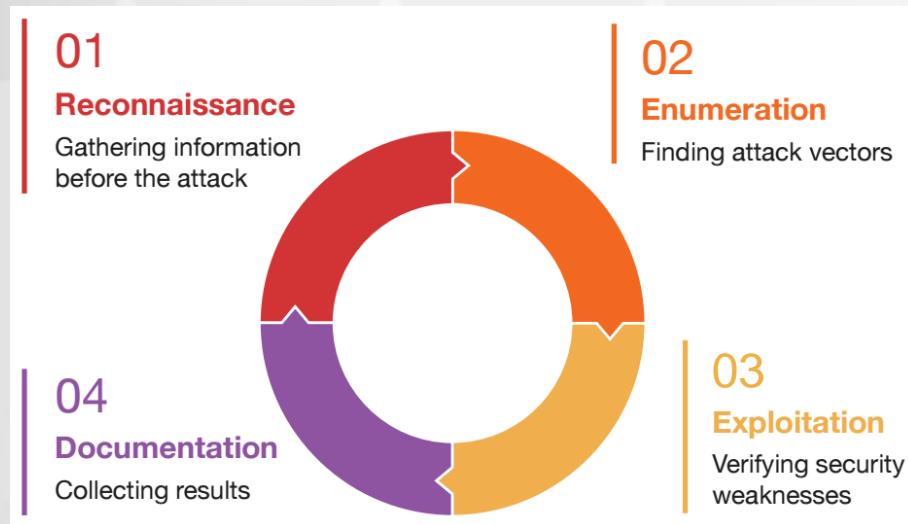
Who are these people?

- All ages
- All levels of experience
- All over the world
- Users and non-users
- Passionate about security!



The Value of Crowdsourced Testing

Formal Methodologies



The Crowd's Creativity



CONNECT

LEARN

GROW

HOW?



Overview

Pre-Launch

- Scope
- Focus
- Exclusions
- Environment
- Access



Post-Launch

- Managing Expectations
- Communicating Effectively
- Defining a Vulnerability Rating Taxonomy

But you never mentioned paying rewards!

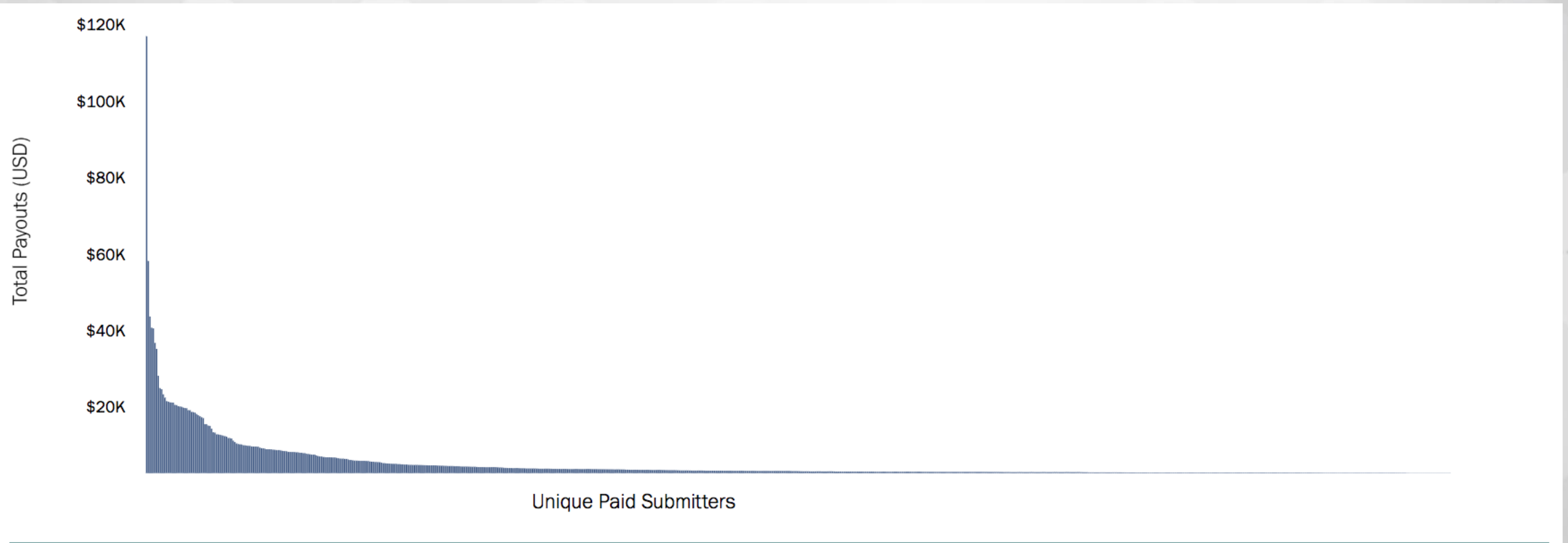


Figure 20: Distribution of cumulative payments cross individual paid submitters.

“Touch the code, pay the bug.”



CONNECT.

LEARN.

GROW.

PRE-LAUNCH



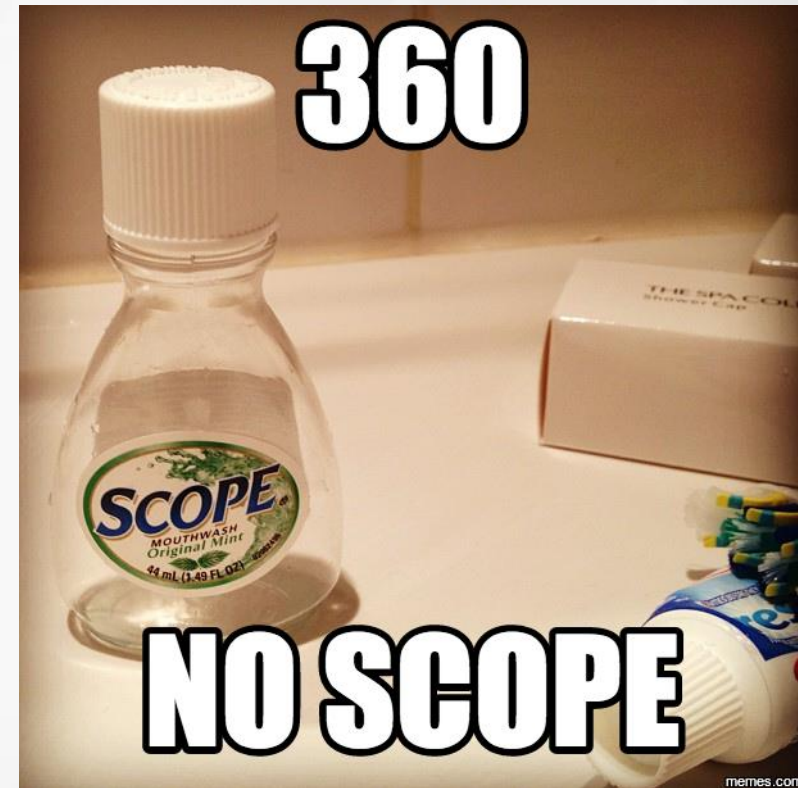
...but first, Step 0

- Basic resources and requirements
 - A full-time resource
 - Escalation policies
 - Organization-wide awareness

Find all of
the dank
memes for
your slides

Scope

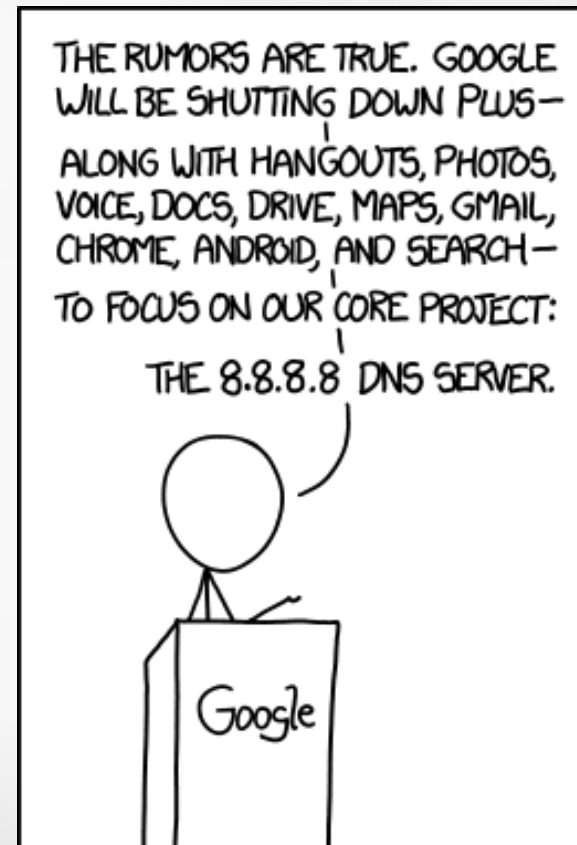
- Scope defines the researcher's universe
 - Leave nothing open to interpretation
 - Understand your attack surface
 - Recognize the path of least resistance



[Source \(Flickr\)](#)

Focus

- You might care about specific:
 - Targets
 - Vulnerability Types
 - Functionality (e.g. payment processing)
- How?
 - Incentives



[Source \(xkcd\)](#)

Exclusions

- You might *not* care about:
 - (Low-impact) “low hanging fruit”
 - Intended functionality
 - Known issues
 - Accepted risks
 - Issues based on pivoting



[Source \(Meme Generator\)](#)

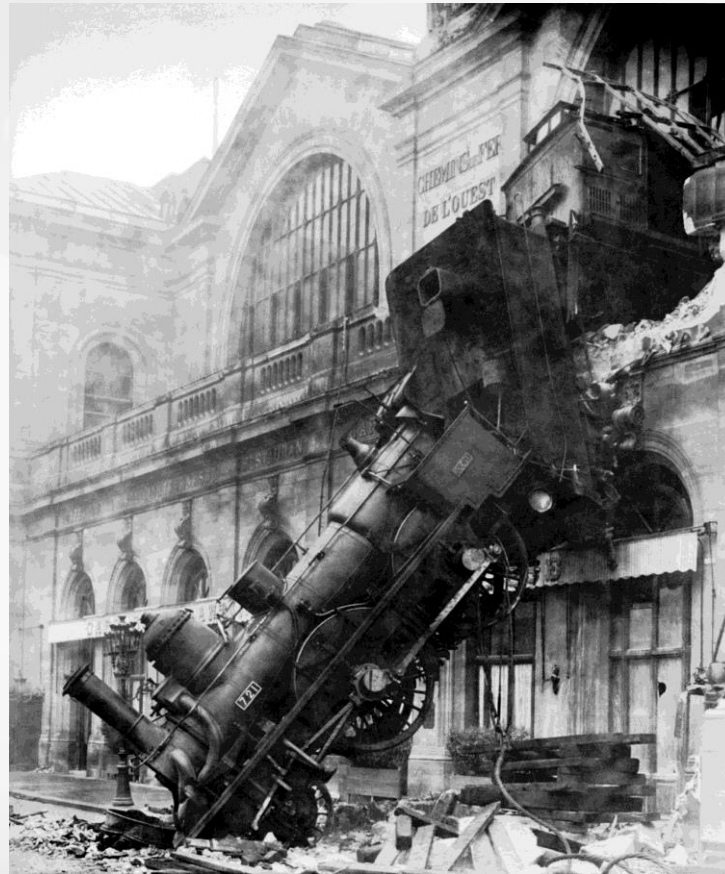
Environment

- Production vs. staging
- Make sure it can stand up to testing!
 - Scanners
 - Contact forms
 - Pentesting requests
- Special bounty types
- Researcher environments



[Source \(Twitter @PokemonGoApp\)](#)

This is what a shared environment looks like...



Access

- Easier = better
- Provide adequate resources for success
 - E.g. sandbox credit cards
- No shared credentials



[Source \(Demotivation\)](#)

Remember...



CONNECT

LEARN

GROW

POST-LAUNCH

Manage Expectations



[Source \(Jane Donald\)](#)

Manage Expectations



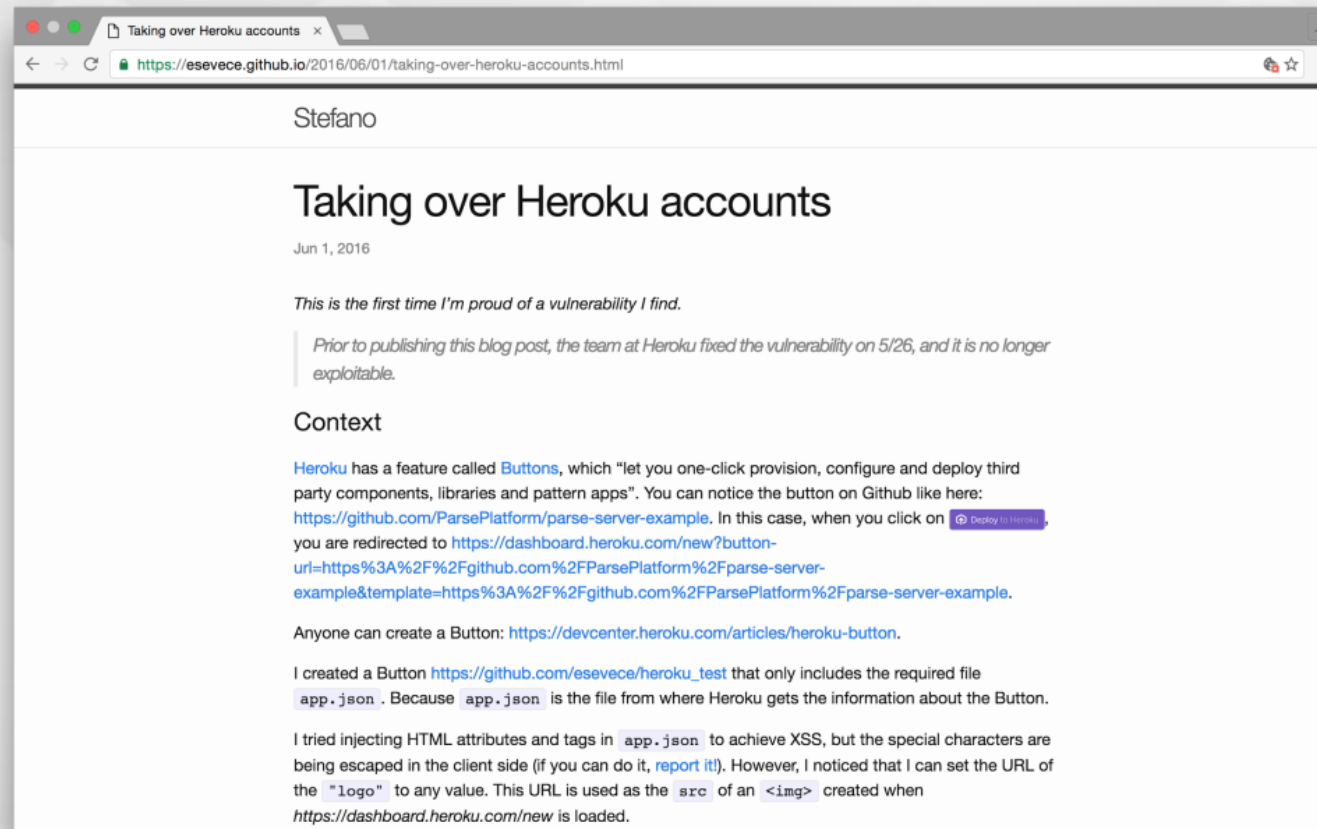
[Source \(SoJo 104.9\)](#)

Communication is Key

- Researchers like:
 - Concise, unambiguous responses
 - ESL
 - Short response time
 - Predictable reward time
- Stay on top of these issues!
- Public disclosure?



Coordinated Disclosure



[Source \(Stefano Vettorazzi\)](#)

Define a Vulnerability Rating Taxonomy (VRT)

- For program owners:
 - Speeds up triage process
 - Track your organization's security posture
 - Arrive at a reward amount more quickly
- For researchers:
 - Focus on high-value bugs
 - Avoid wasting time on non-rewardable bugs
 - Alongside brief, helps build trust



CONNECT

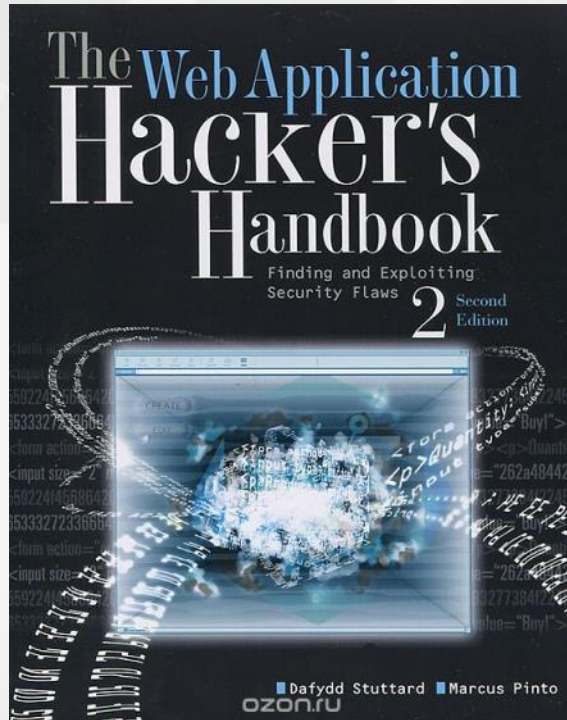
LEARN

GROW

THINK LIKE A RESEARCHER



The Regular Methodologies



[Source \(Amazon\)](#)



[Source \(OWASP\)](#)

The Bughunter's Methodology

- Identify roads less traveled
 - Acquisitions (define the rules)
 - Functionality changes or redesigns
 - Mobile websites or apps
- Think like a researcher
 - Wikipedia (acquisitions)
 - Google dorks, recon-ng, altdns, etc.
 - [jhaddix/domain](#) (enumall)
 - [ChrisTruncer/EyeWitness](#)
 - Researchers' Blogs
 - And many more we can't fit into this talk!



[Source \(Meme Generator\)](#)

CONNECT

LEARN

GROW

A GOOD REPORT





Insecure Direct Object Reference - Multiple Billing API Endpoints

100% · Researcher001 · 01/01/2020

The payment billing endpoint returns customer billing information (<cool stuff you can use to steal money>, etc.). The <flux capacitor> ID is used to request the information. By iterating through different <flux capacitor> IDs, I was able to view billing information for other customers.

Reference Number <some reference number>

Original caption Insecure Direct Object Reference - Billing Detail Disclosure

Bug Type Bug/Other

XSS Location URL Empty

Affected Parameter <flux capacitor id> ID

Affected Users AUTHENTICATED

Attack String Empty

Browser Empty

Bug URL <some url>

Device Empty

HTTP Request

```
Host:<some url>
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:<some url>
Connection: close

-----

HTTP/1.1 200 OK
Cache-Control:<stuff>
Content-Type: application/json; charset=UTF-8
```

Method of Finding manual

Platform Empty

Platform Version Empty

Proof of Concept Empty

Replication Steps

1. Configure your browser to use an intercepting proxy such as Burp or monitor the request using Chrome/Firefox developer tools.
2. Login to the web application and browse to the billing information page
3. Capture the request to the billing information endpoint and send it to Repeater or Intruder
4. Modify the request to attempt to enumerate additional <flux capacitor> IDs and observe the billing information in the response.

Tools Used Burp, Intruder

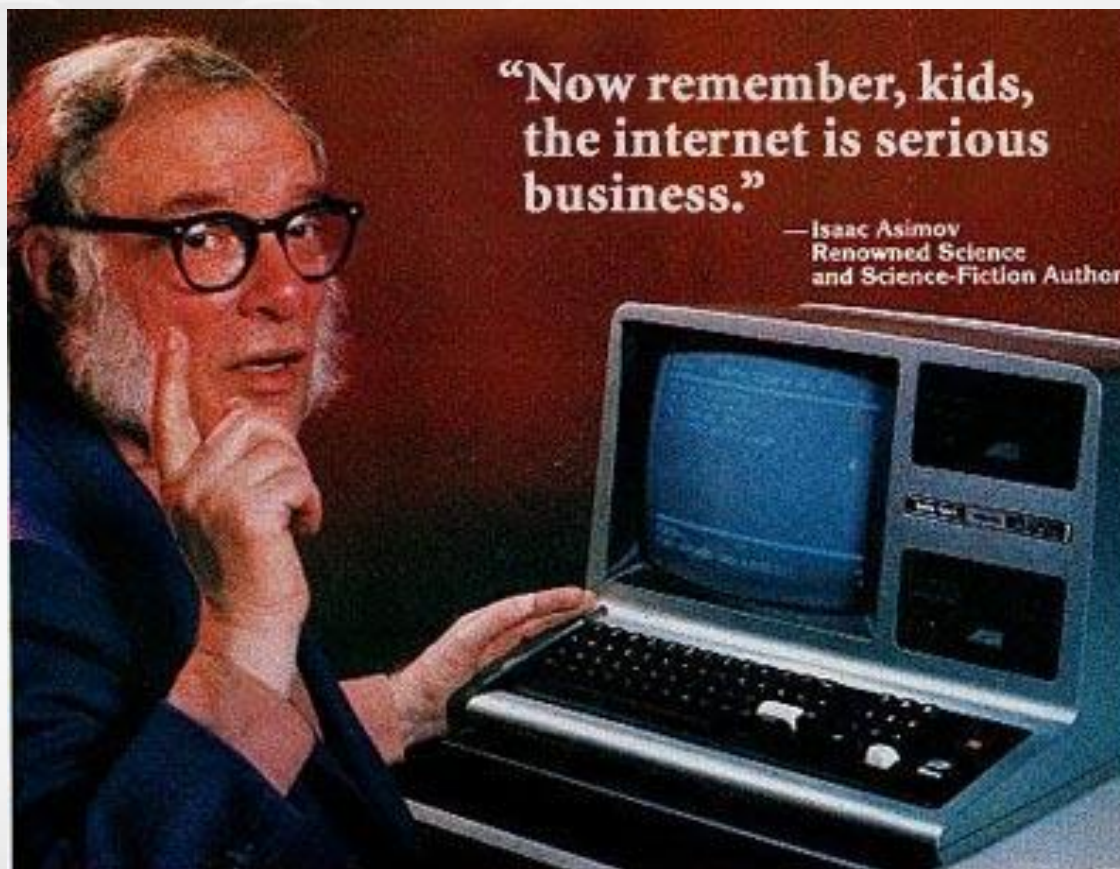
CONNECT

LEARN

GROW

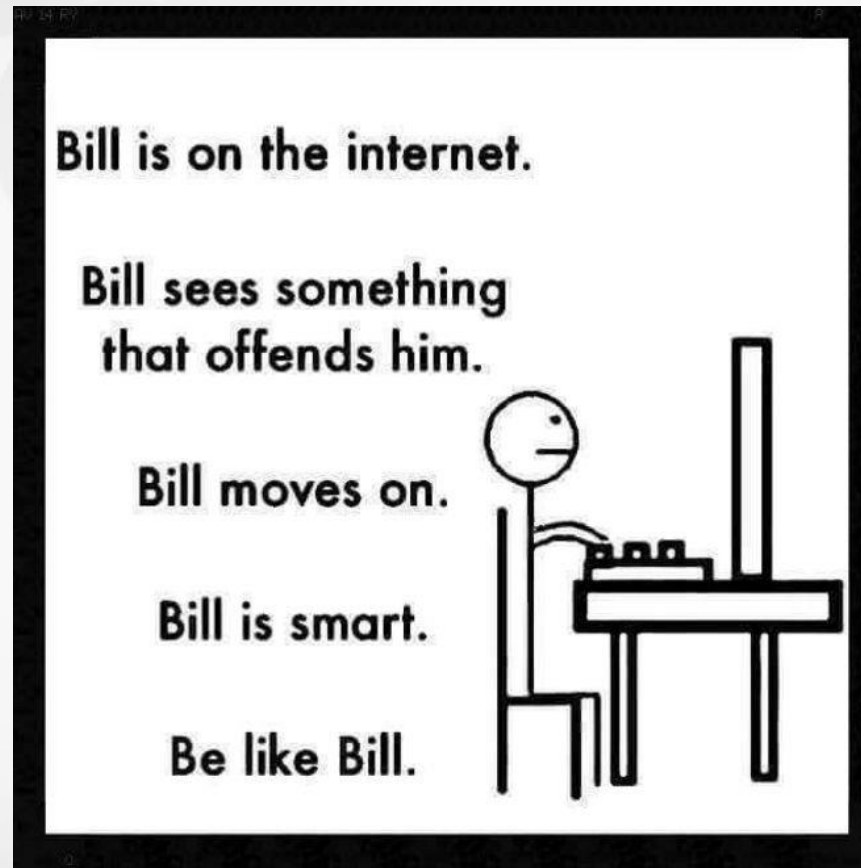
FINAL TIPS

Consider the business impact!



[Source \(Know Your Meme\)](#)

Remember what it's all about.



[Source \(BBC\)](#)

Case Study: Instructure

CONNECT

	2013 (Pentest)	2014 (Bug Bounty)	2015 (Bug Bounty)
Critical	0	0	0
High	1	25	3
Medium	1	8	2
Low	2	16	5

[Source \(Instructure\)](#)

CONNECT

LEARN

GO!!!



[Source \(Stack Exchange\)](#)

Grant McCracken
grant@bugcrowd.com

Daniel Trauner
dan@bugcrowd.com

