Scaling Security, Building Relationships, Building the Future

sec4dev 8th September 2022

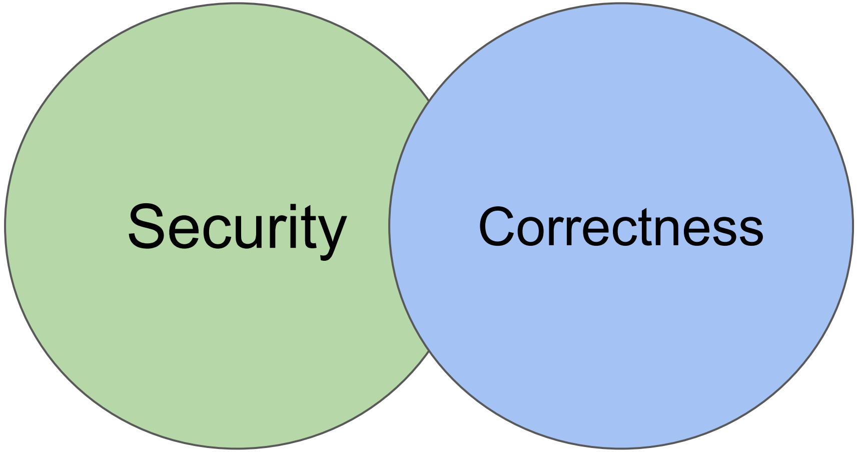
Clint Gibler

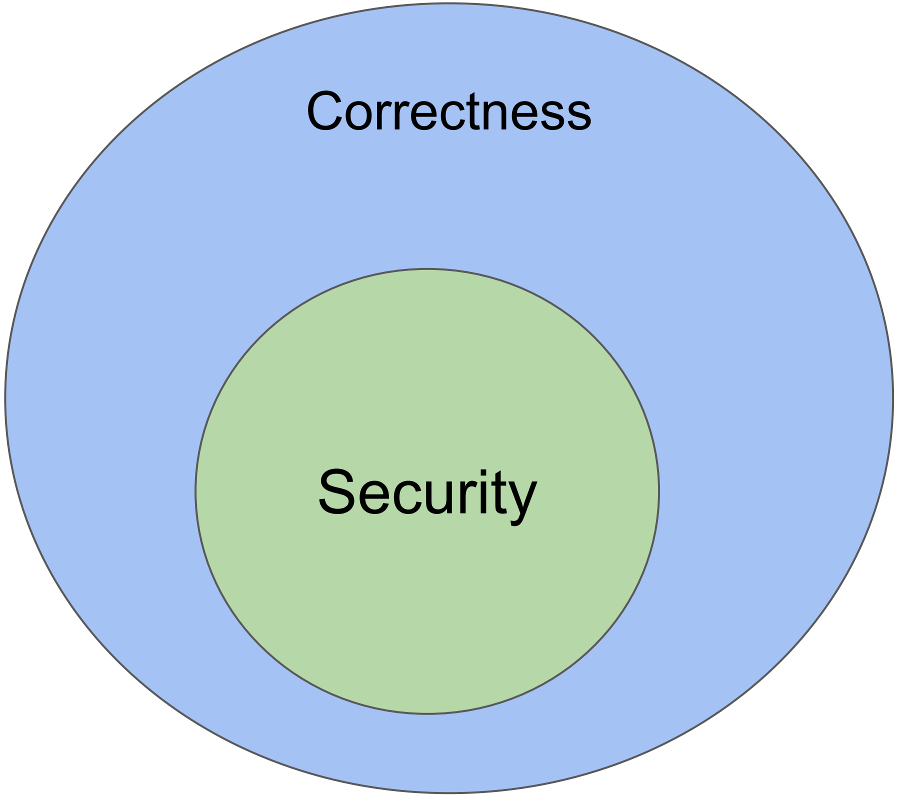
@clintgibler

Tldrsec.com

THE FUTURE
OF SOFTWARE
DEVELOPMENT
DEVELOPMENT
SECURITY

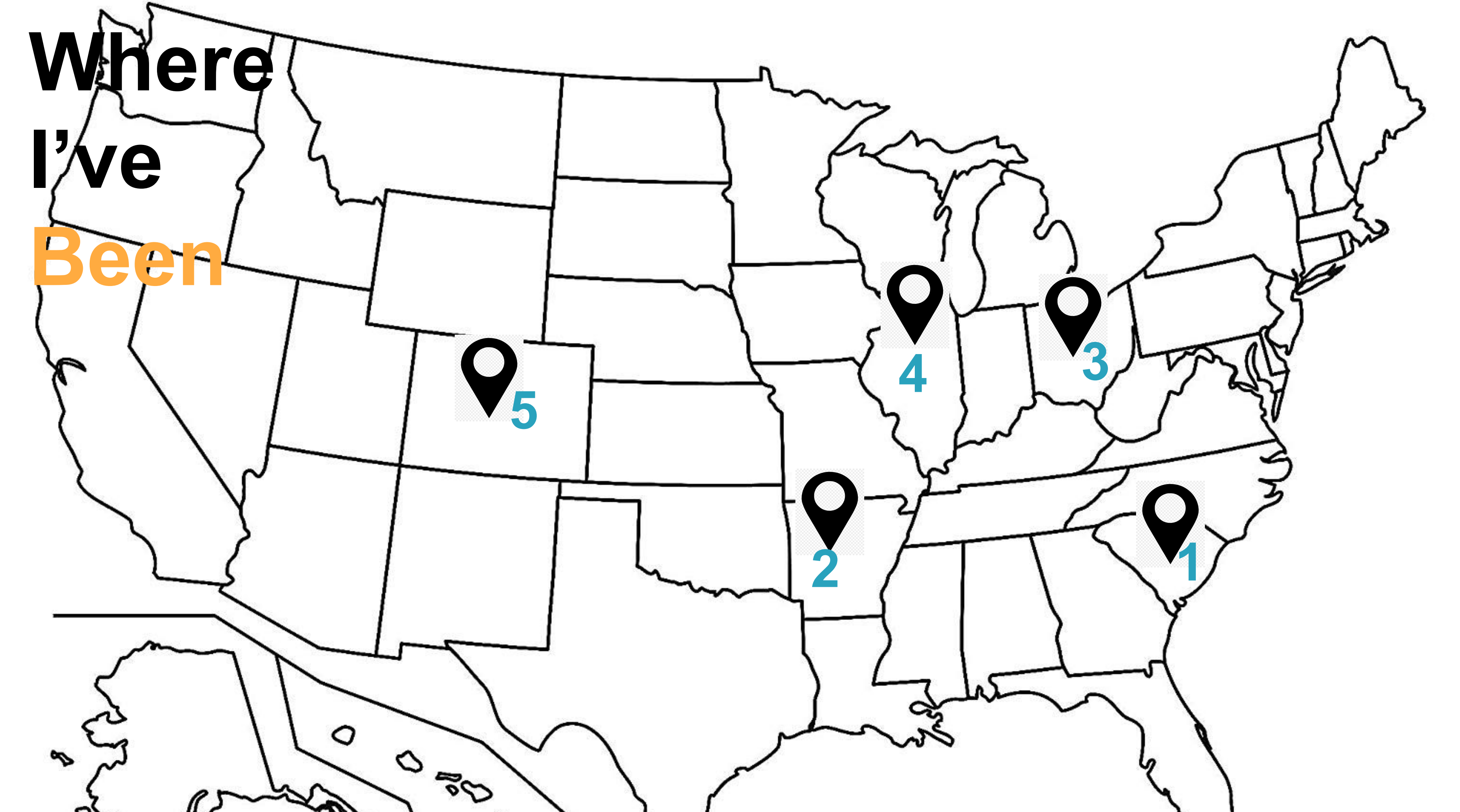
Security and Correctness





- High quality software <--> secure software
- The same approach and mindset can improve quality **and** security
  - → Faster dev velocity!

Where I've Been



Where I've Been

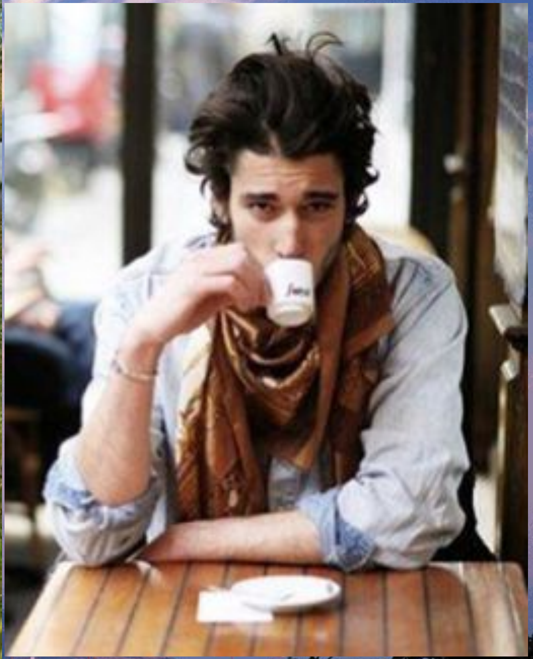
AndroidLeaks: Automatically Detecting Potential Privacy Leaks In Android Applications on a Large Scale

Clint Gibler[1], Jonathan Crussell[1,2], Jeremy Erickson[1,2], and Hao Chen[1]

TRUST 2012
5th International Conference on Trust & Trustworthy Computing
13-15 June 2012, Vienna, Austria

© Grantus Osborne

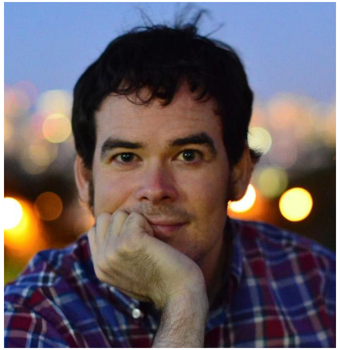
Where I Am



https://www.lemosfarm.com/goat-yoga

whois?

Clint Gibler
Head of Security Research @ r2c
Formerly: NCC Group, UC Davis PhD
@clintgibler
tldrsec.com

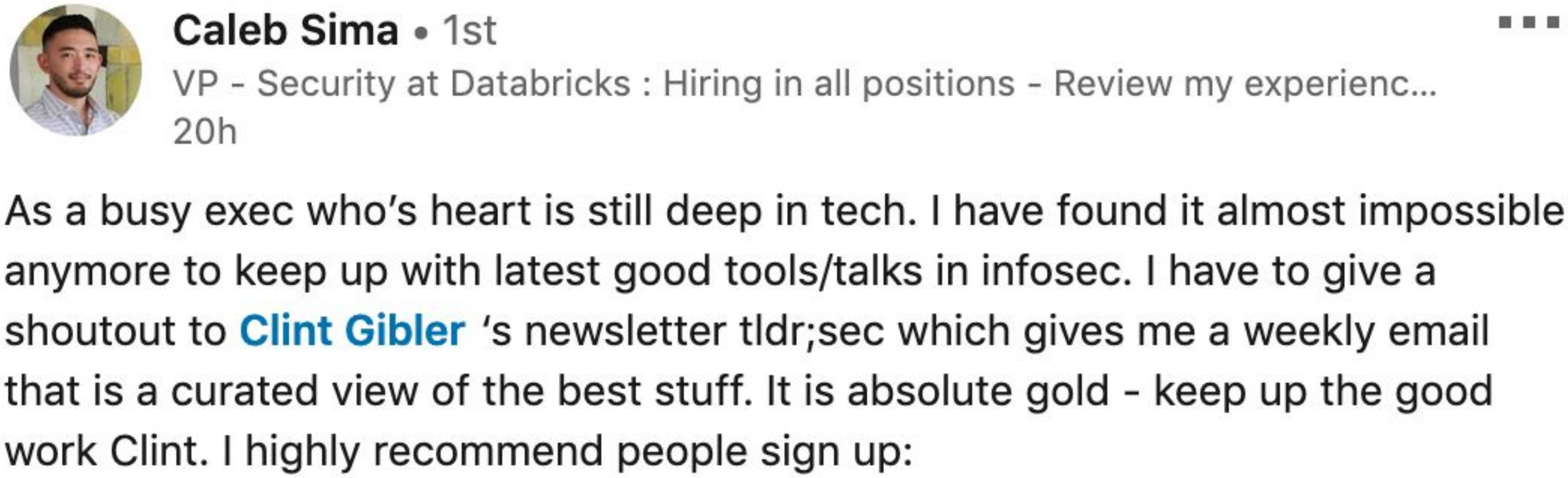


r2c:
We're an SF based static analysis startup on a mission to profoundly improve software security and reliability.

Biō
denns BioMa

Tl;dr sec Newsletter - https://tldrsec.com

- Talk summaries | Tools & Resources links | Original research



Caleb Sima • 1st

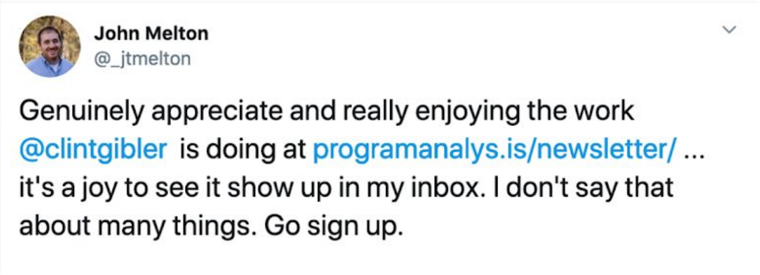
VP - Security at Databricks : Hiring in all positions - Review my experienc...

20h

As a busy exec who's heart is still deep in tech. I have found it almost impossible anymore to keep up with latest good tools/talks in infosec. I have to give a shoutout to Clint Gibler 's newsletter tldr;sec which gives me a weekly email that is a curated view of the best stuff. It is absolute gold - keep up the good work Clint. I highly recommend people sign up:

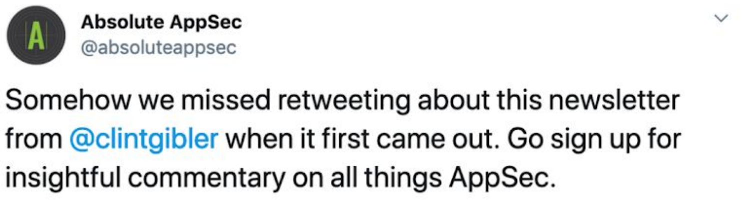
Tl;dr sec Newsletter - https://tldrsec.com

- Talk summaries | Tools & Resources links | Original research



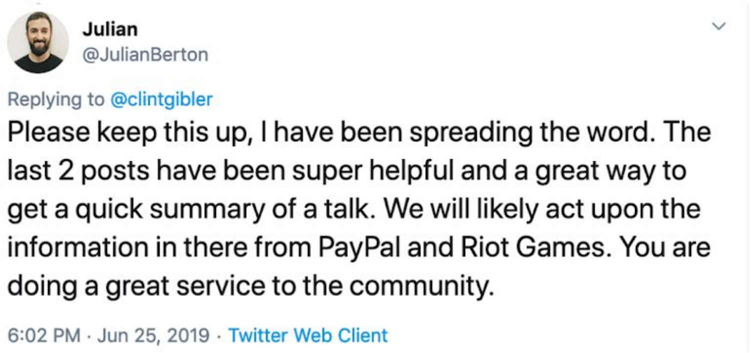
John Melton
@_jtmelton

Genuinely appreciate and really enjoying the work @clintgibler is doing at programanalys.is/newsletter/ ... it's a joy to see it show up in my inbox. I don't say that about many things. Go sign up.



Absolute AppSec
@absoluteappsec

Somehow we missed retweeting about this newsletter from @clintgibler when it first came out. Go sign up for insightful commentary on all things AppSec.

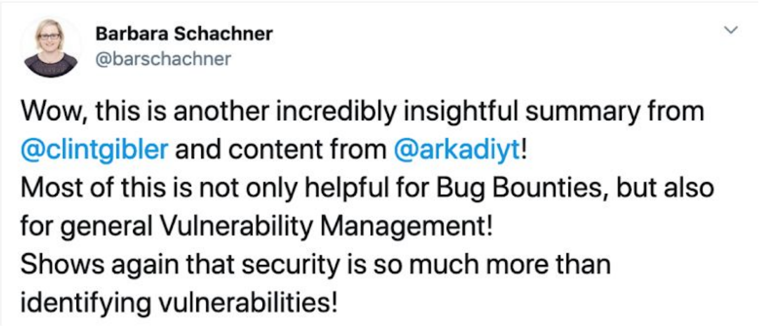


Julian
@JulianBerton

Replying to @clintgibler

Please keep this up, I have been spreading the word. The last 2 posts have been super helpful and a great way to get a quick summary of a talk. We will likely act upon the information in there from PayPal and Riot Games. You are doing a great service to the community.

6:02 PM · Jun 25, 2019 · Twitter Web Client



Barbara Schachner
@barschachner

Wow, this is another incredibly insightful summary from @clintgibler and content from @arkadiyt!
Most of this is not only helpful for Bug Bounties, but also for general Vulnerability Management!
Shows again that security is so much more than identifying vulnerabilities!

Who’s here?

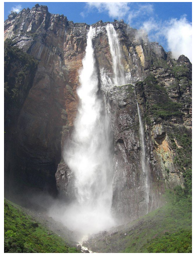
Agenda

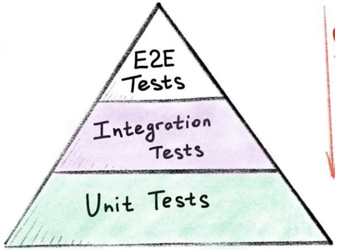
- ~~Whoami~~
- ~~Whoareyou~~
- Development and Security - History and Changes
- Why can security people be grumpy sometimes?
- Building Better Developer <> Security Relationships
- Things I’m Excited About

A Very Abridged History

Development

QA team





Time

manually

Security

Why can security people be grumpy sometimes?

Security New Grad

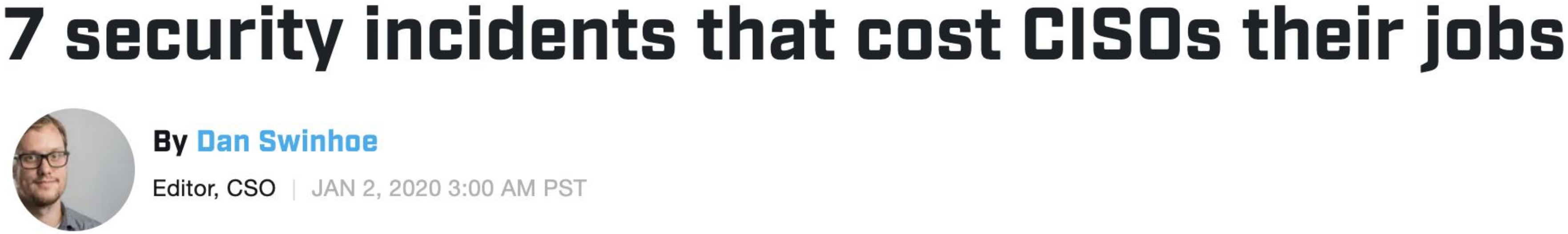
3 years into industry

Cost Center vs Business Enabler

Makes money

Costs money

Blamed When Things Go Wrong



7 security incidents that cost CISOs their jobs

By Dan Swinhoe

Editor, CSO | JAN 2, 2020 3:00 AM PST

- Warn the business about potential or existing risks
- “Cool story, but we have features to ship and product deadlines/revenue goals”
- Faces repercussions because of the results of other people’s actions

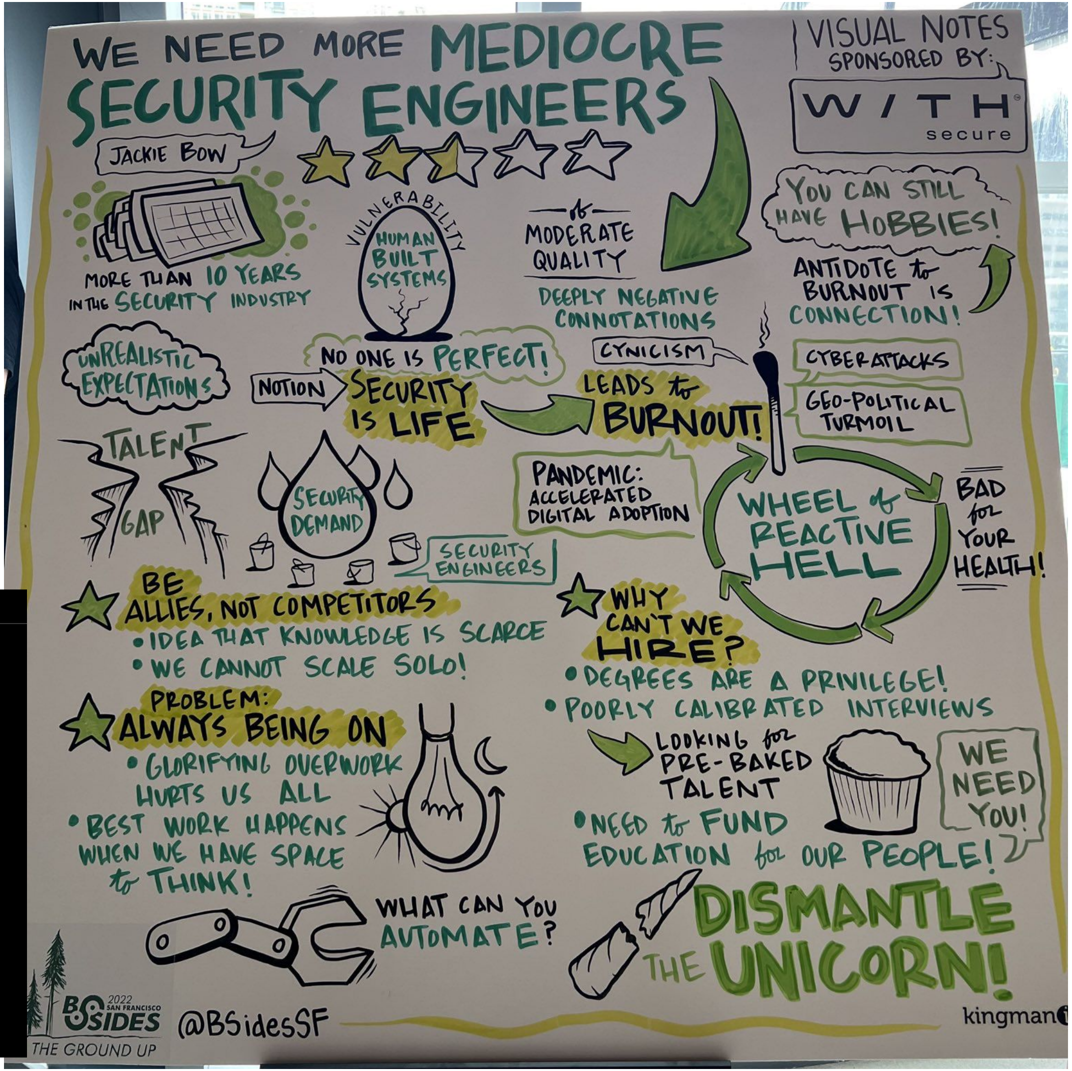
Burnout, High Standards

We Need More Mediocre Security Engineers
By Jackie Bow

Building sustainable security programs
By Astha Singhal

OUTLINE

- CONTRIBUTING FACTORS
- ORGANIZATIONAL CULTURE
- RISK PERSPECTIVE
- STRATEGIC PROGRAM FOCUS
- STAKEHOLDER AND LEADERSHIP ALIGNMENT
- KEY TAKEAWAYS



WE NEED MORE MEDIOCRE SECURITY ENGINEERS
JACKIE BOW
VISUAL NOTES SPONSORED BY: WITH secure
MORE THAN 10 YEARS IN THE SECURITY INDUSTRY
VULNERABILITY
HUMAN BUILT SYSTEMS
of MODERATE QUALITY
DEEPLY NEGATIVE CONNOTATIONS
YOU CAN STILL HAVE HOBBIES!
ANTIDOTE to BURNOUT IS CONNECTION!
UNREALISTIC EXPECTATIONS
NO ONE IS PERFECT!
NOTION
SECURITY IS LIFE
CYNICISM
LEADS to BURNOUT!
CYBERATTACKS
GEO-POLITICAL TURMOIL
TALENT GAP
SECURITY DEMAND
SECURITY ENGINEERS
PANDEMIC: ACCELERATED DIGITAL ADOPTION
WHEEL of REACTIVE HELL
BAD for YOUR HEALTH!
BE ALLIES, NOT COMPETITORS
• IDEA THAT KNOWLEDGE IS SCARCE
• WE CANNOT SCALE SOLO!
WHY CAN'T WE HIRE?
• DEGREES ARE A PRIVILEGE!
• POORLY CALIBRATED INTERVIEWS
PROBLEM: ALWAYS BEING ON
• GLORIFYING OVERWORK HURTS US ALL
• BEST WORK HAPPENS WHEN WE HAVE SPACE to THINK!
LOOKING for PRE-BAKED TALENT
WE NEED YOU!
• NEED to FUND EDUCATION for OUR PEOPLE!
WHAT CAN YOU AUTOMATE?
DISMANTLE THE UNICORN!
2022 SAN FRANCISCO BSIDES THE GROUND UP
@BSidesSF
kingman

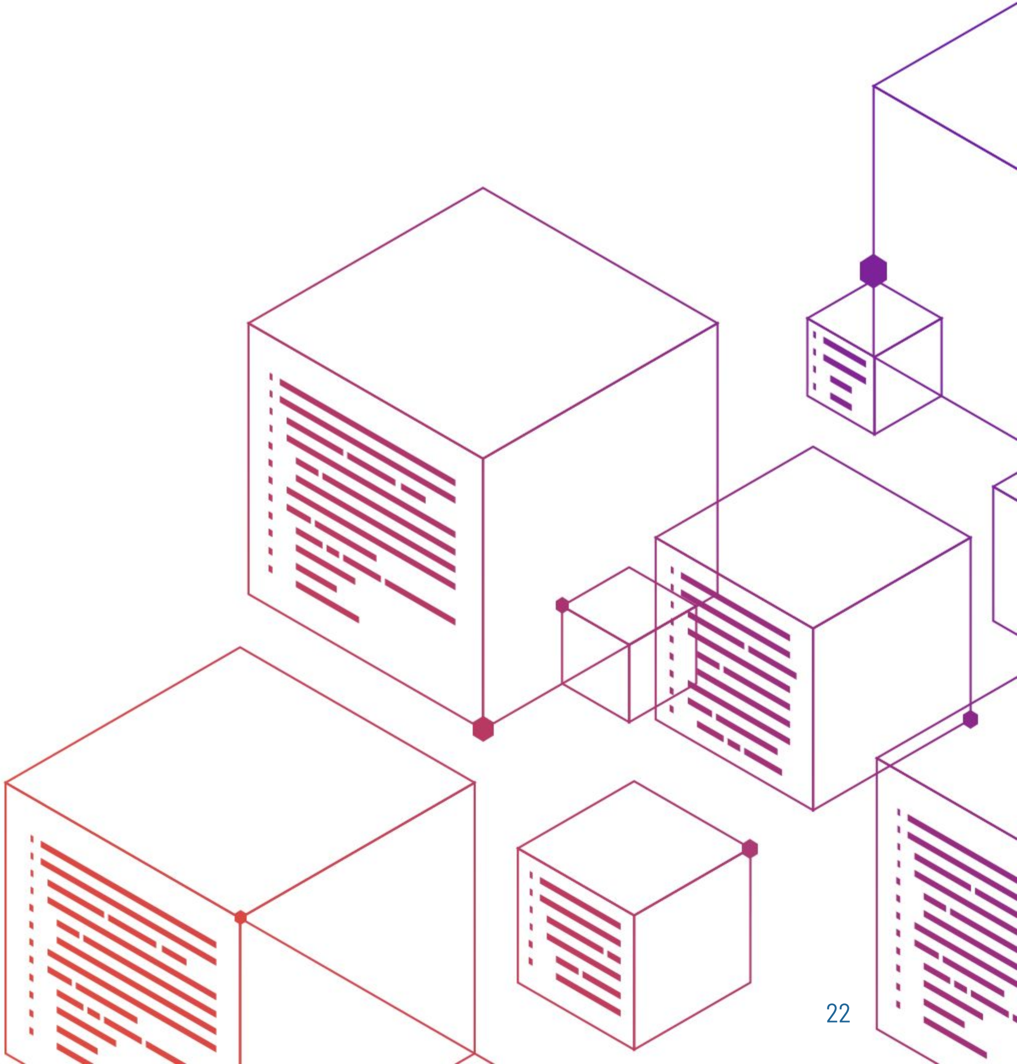
Best Case: Nothing Happens

- When security team is doing well → nothing happens
- Have to get every single thing right, attacker needs only 1 way in
- Seeking perfection can lead to burnout
- Can be demoralizing

Breakers, not Builders (Historically)

- In the past, security focused on finding vulnerabilities, **not** fixing them
- Network security or sysadmin background → Didn't write/read software
- Hard for security to understand/empathize with dev processes, grok how much effort an ask might involve

Building Better Dev 🤝 Security Relationships



Cross-Team Embeds

- Security - spend a month or quarter working on an engineering team (and vice versa)
- Learn how code is written and shipped to production
- Understand when asks are going to be prohibitively difficult to accomplish
- Build allies, empathy, and trust

DAY 45: I HAVE EARNED THE GERMANS' TRUST
THEY STILL DO NOT REALIZE I AM BEAR

Ask For Help

- Guaranteed to make security happy
- Security: Make it easy to ask
  - Clear channels, be kind

Build a Paved Road

- Consistent code is high quality (and secure) code
- Standardize on libraries, tooling, deployment pipelines, etc. that make the **best** way the **easiest** way
  - Performance, logging/visibility, security
  - Building blocks or functionality that can be easily and widely used
- Security and platform engineering team should be best friends
- "**Hitch your security wagon to developer productivity**" - Patrick Thomas, Astha Singhal
- More on this later

Build Shared Capabilities

Seek mutual wins - tools, libraries, infrastructure that help dev **and** security

Asset Inventory

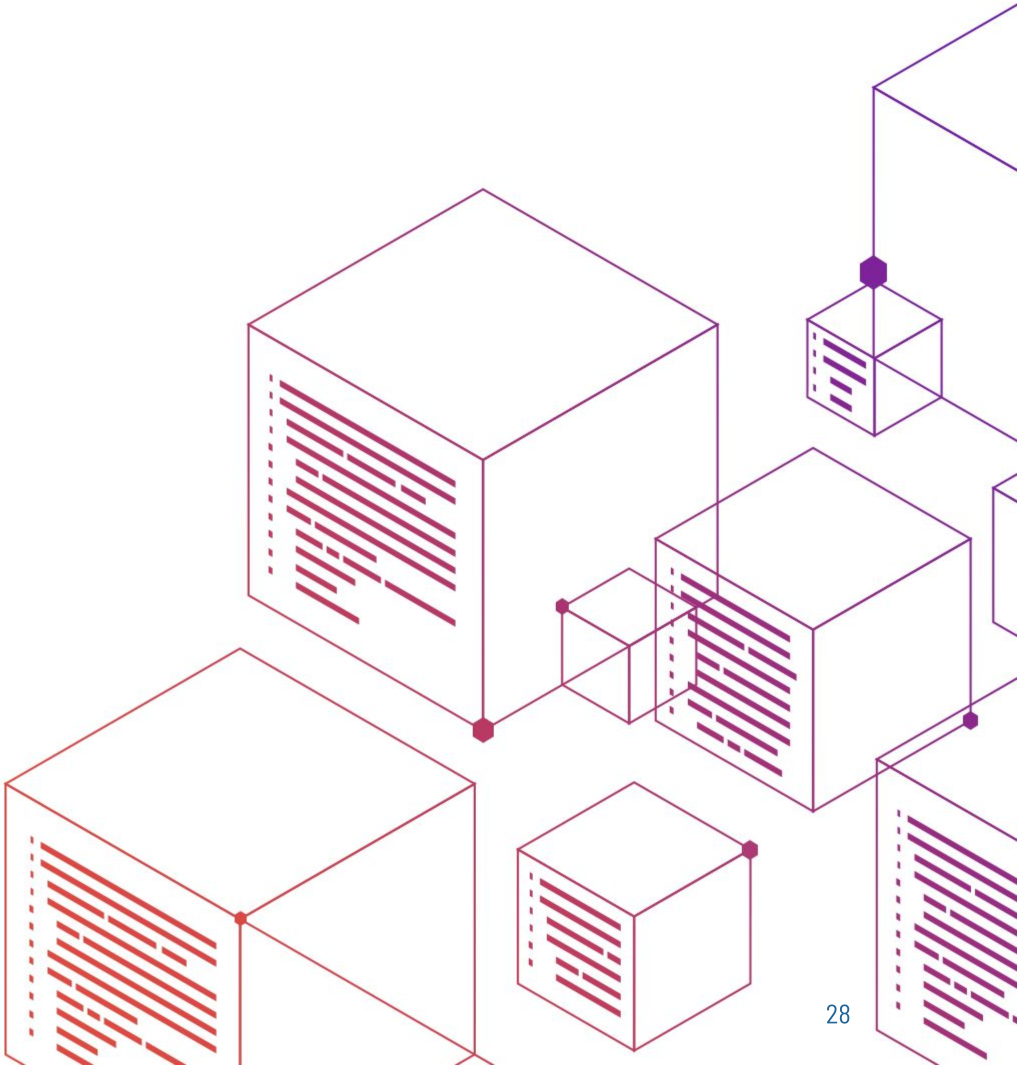
- Useful for security - need to know what we have to protect it
- Also DevOps/SRE (troubleshooting), finance (billing), engineering (what do we have, where is it?)

Code Scanning

- Ensuring code standards at scale
- Large scale code refactoring (e.g. upgrading from deprecated APIs)
- Helping onboard new developers

“My single best, most effective security spend...”
- Zane Lackey

Things I’m Excited About



Things I’m Excited About

- “Customer-centric” security teams
- “Guardrails, not Gatekeepers”
- Ecosystems getting better (web frameworks, browser security properties, memory safe languages, ...)
- Secure Guardrails

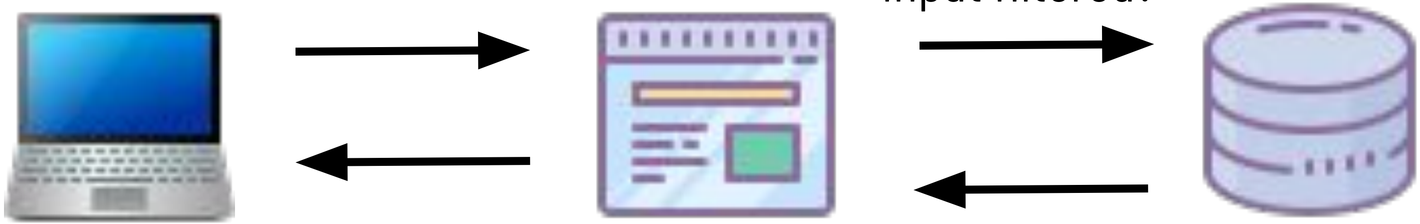
🤔 A Different Way to Approach Security

- Killing bug classes: scalable, systematic, long-term wins
- Enabling developers to move fast *and* securely
    - Security team as business enablers, not another point of friction

Quiz: Does this app have XSS?

Icons by Icons8

Quiz: Does this app have XSS?



What does user control?
Structure of data?

Input filtered?

How is it stored?
(field types,
constraints)

DB type?

Context?

- HTML
- HTML attribute
- JavaScript
- ...

Data processed before sent to user?

Icons by Icons8

Quiz: Does this app have XSS?

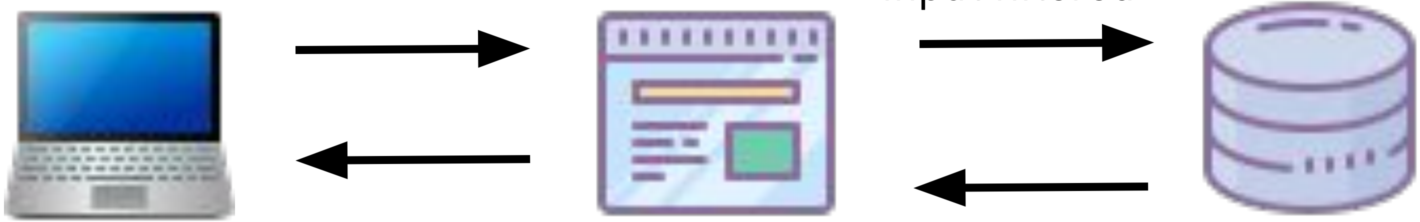
Guardrail: Frontend is React, banned dangerouslySetInnerHTML

What does user control?
Structure of data?

Input filtered?

How is it stored?
(field types,
constraints)

DB type?



Context?

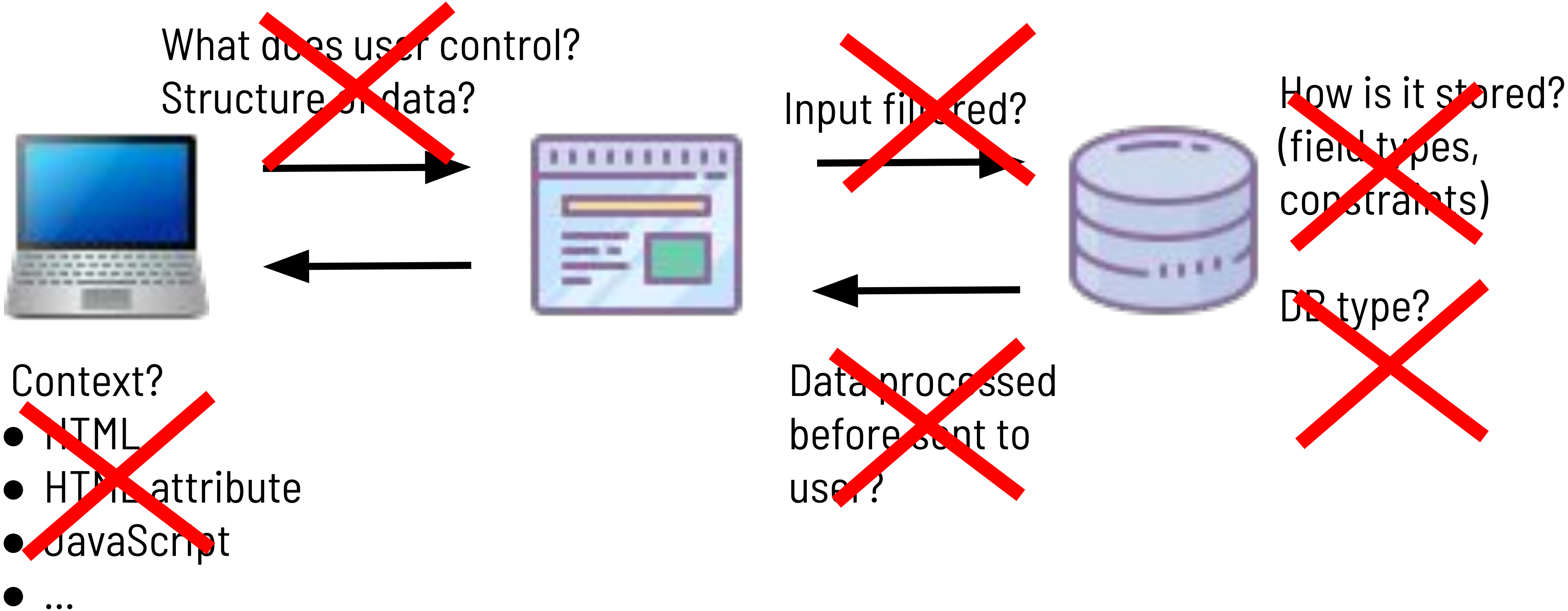
- HTML
- HTML attribute
- JavaScript
- ...

Data processed
before sent to
user?

Icons by Icons8

Quiz: Does this app have XSS?

Guardrail: Frontend is React, banned dangerouslySetInnerHTML



~~What does user control?~~
~~Structure of data?~~

~~Input filtered?~~

~~How is it stored?~~
~~(field types,~~
~~constraints)~~

~~DB type?~~

~~Data processed before sent to user?~~

Context?

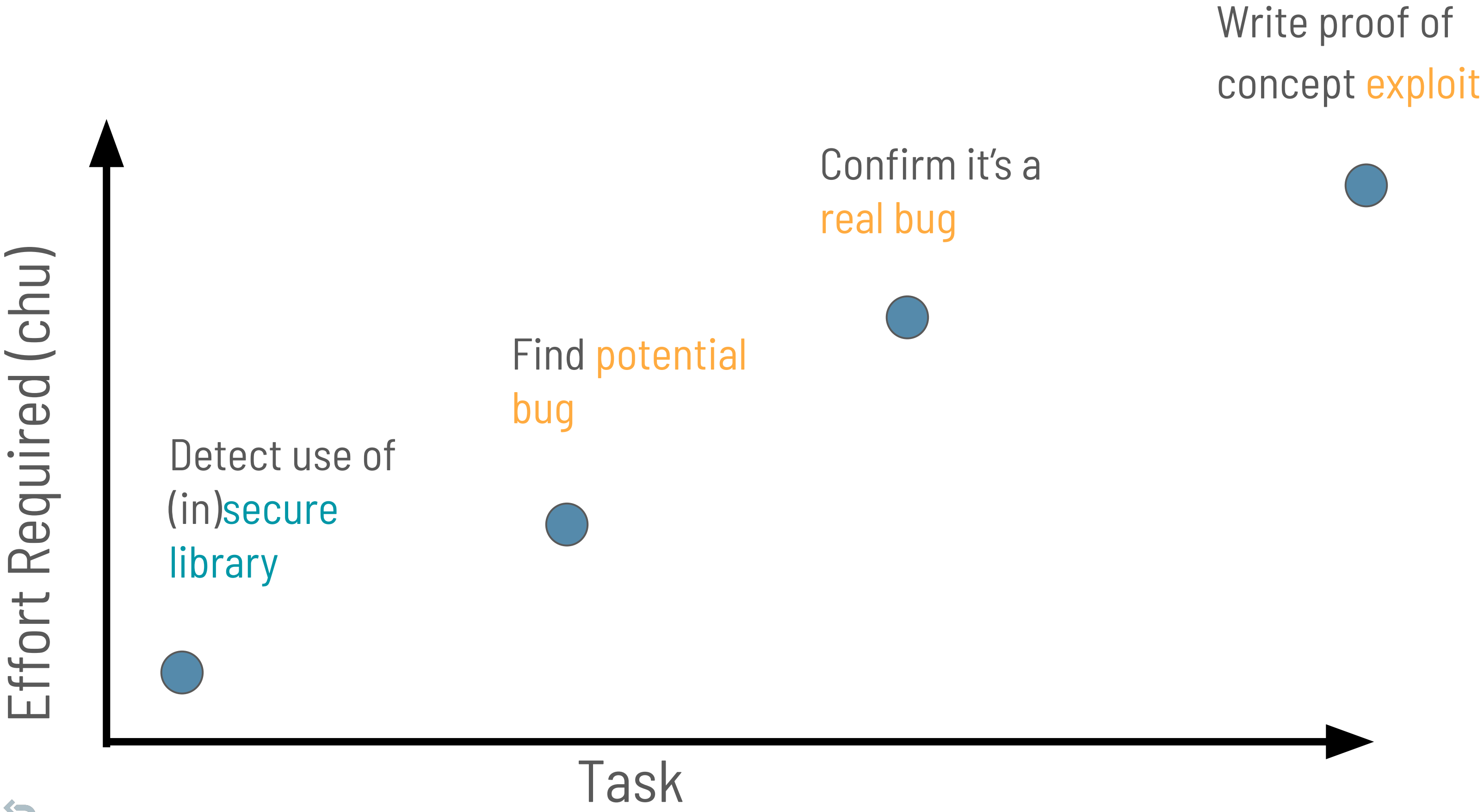
- ~~HTML~~
- ~~HTML attribute~~
- ~~JavaScript~~
- ...

Icons by Icons8

Let’s Solve the “Easy” Version of the Problem

- This app could have been incredibly complex, with millions of LOC
- With some strong secure defaults, we significantly reduced its risk

Task vs Effort Required



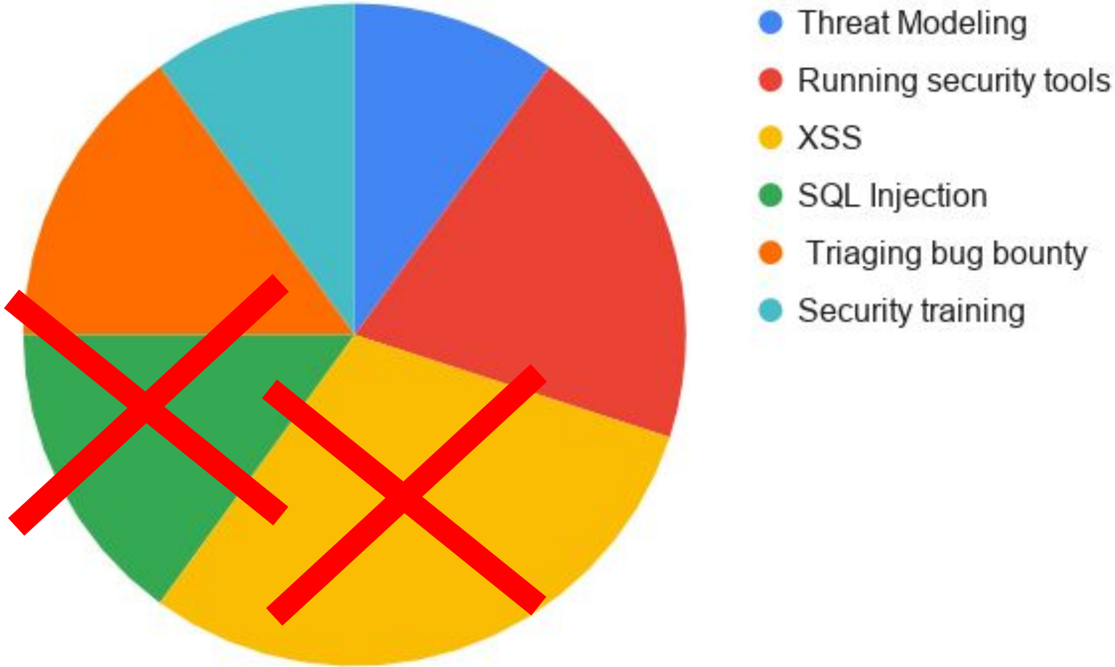
Detect use of (in)secure library

Find potential bug

Confirm it's a real bug

Write proof of concept exploit

Compounding Effects of Killing Bug Classes



Detecting (lack of) use of
secure defaults

is **much easier** than

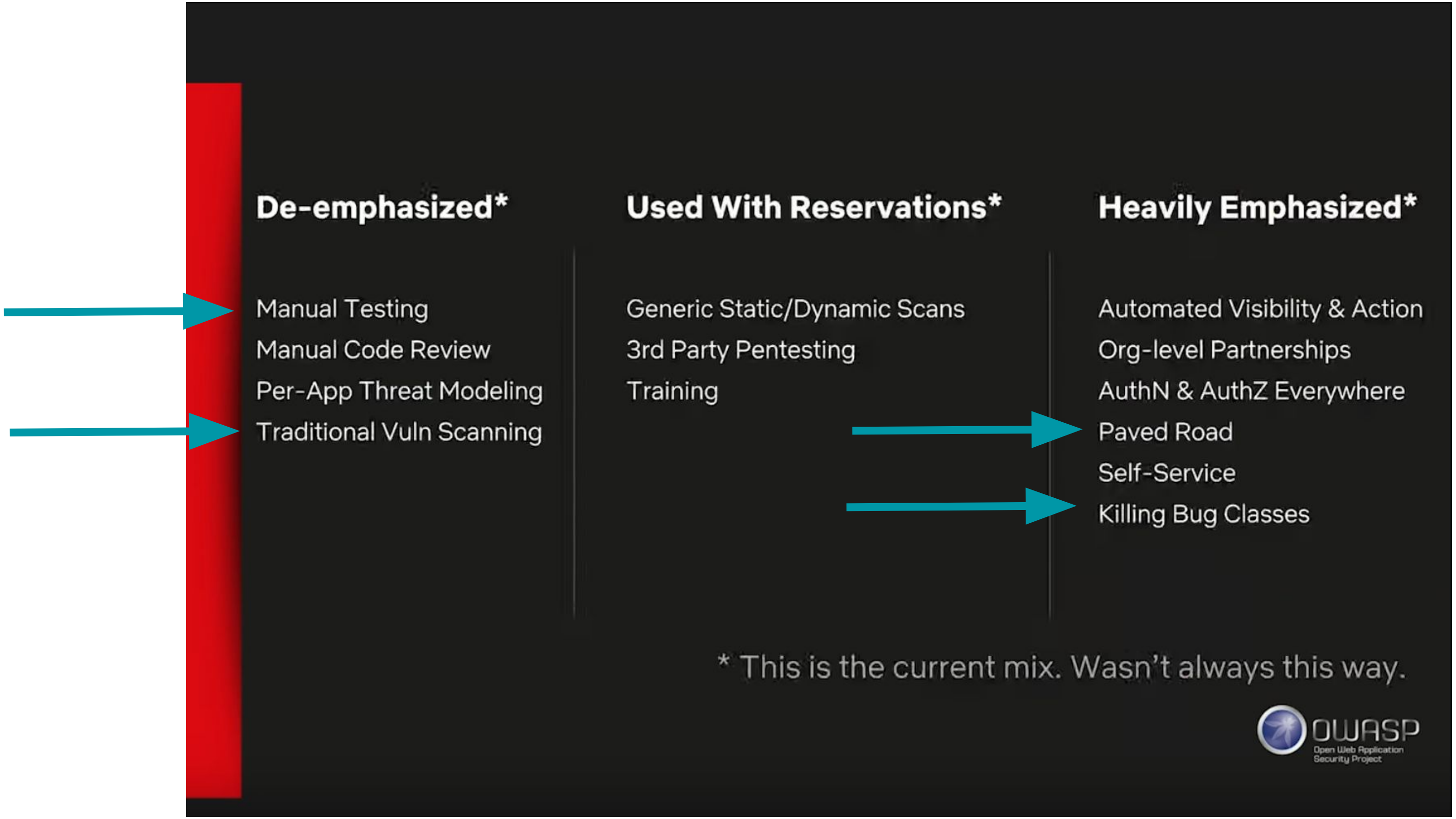
finding **bugs**

Your Internal Dialogue?

- “All you’ve shown me is some hand-wavy diagrams”
- The security industry has focused on bug finding for decades
  - SAST, DAST, pen tests, bug bounty

We Come Bearing Gifts: Enabling Prod Security w/ Culture & Cloud

AppSec Cali '18, Patrick Thomas, Astha Singhal

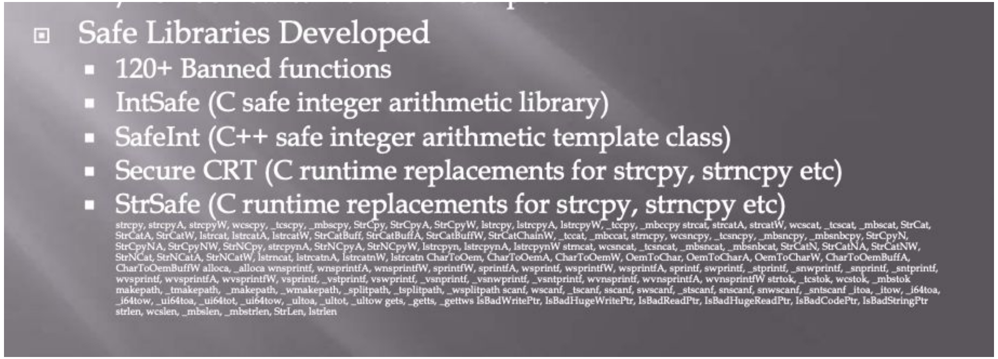


De-emphasized*	Used With Reservations*	Heavily Emphasized*
Manual Testing	Generic Static/Dynamic Scans	Automated Visibility & Action
Manual Code Review	3rd Party Pentesting	Org-level Partnerships
Per-App Threat Modeling	Training	AuthN & AuthZ Everywhere
Traditional Vuln Scanning		Paved Road
		Self-Service
		Killing Bug Classes

* This is the current mix. Wasn't always this way.

How Valuable Can Banning Functions Be?

41% of vulnerability reduction from XP → Vista from banning *strcpy* and friends



- Safe Libraries Developed
  - 120+ Banned functions
  - IntSafe (C safe integer arithmetic library)
  - SafeInt (C++ safe integer arithmetic template class)
  - Secure CRT (C runtime replacements for strcpy, strncpy etc)
  - StrSafe (C runtime replacements for strcpy, strncpy etc)

strcpy, strcpyA, strcpyW, wcscpy, _tcscpy, _mbscpy, StrCpy, StrCpyA, StrCpyW, lstrcpy, lstrcpyA, lstrcpyW, _tccpy, _mbccpy strcat, strcatA, strcatW, wcscat, _tcscat, _mbscat, StrCat, StrCatA, StrCatW, lstrcat, lstrcatA, lstrcatW, StrCatBuff, StrCatBuffA, StrCatBuffW, StrCatChainW, _tccat, _mbccat, strncpy, wcsncpy, _tcsncpy, _mbsncpy, _mbsnbcpy, StrCpyN, StrCpyNA, StrCpyNW, StrNCpy, strcpynA, StrNCpyA, StrNCpyW, lstrcpyn, lstrcpynA, lstrcpynW strncat, wcsncat, _tcsncat, _mbsncat, _mbsnbcat, StrCatN, StrCatNA, StrCatNW, StrNCat, StrNCatA, StrNCatW, lstrncat, lstrcatnA, lstrcatnW, lstrcatn CharToOem, CharToOemA, CharToOemW, OemToChar, OemToCharA, OemToCharW, CharToOemBuffA, CharToOemBuffW alloca, _alloca wnsprintf, wnsprintfA, wnsprintfW, sprintfW, sprintfA, wsprintf, wsprintfW, wsprintfA, sprintf, swprintf, _stprintf, _snwprintf, _snprintf, _sntprintf, wvsprintf, wvsprintfA, wvsprintfW, vsprintf, _vstprintf, vswprintf, _vsnprintf, _vsnwprintf, _vsntprintf, wvnsprintf, wvnsprintfA, wvnsprintfW strtok, _tcstok, wcstok, _mbstok makepath, _tmakepath, _makepath, _wmakepath, _splitpath, _tsplitpath, _wsplitpath scanf, wscanf, _tscanf, sscanf, swscanf, _stscanf, snscanf, snwscanf, _sntscanf _itoa, _itow, _i64toa, _i64tow, _ui64toa, _ui64tot, _ui64tow, _ultoa, _ultot, _ultow gets, _getts, _gettws IsBadWritePtr, IsBadHugeWritePtr, IsBadReadPtr, IsBadHugeReadPtr, IsBadCodePtr, IsBadStringPtr strlen, wcslen, _mbslen, _mbstrlen, StrLen, lstrlen

Analysis of 63 buffer-related security bugs that affect Windows XP, Windows Server 2003 or Windows 2000 but not Windows Vista: 82% removed through SDL process

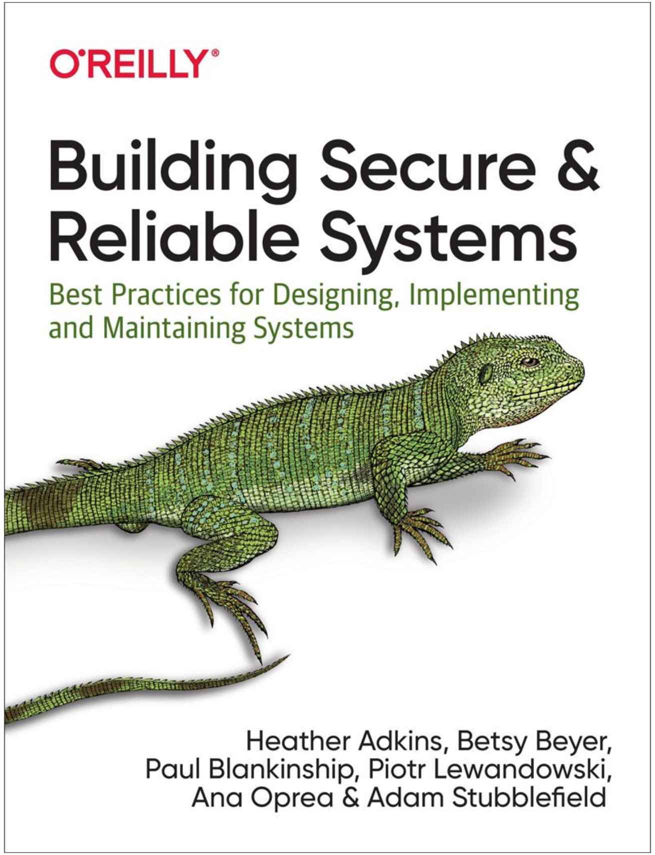
- 27 (43%) found through use of SAL (Annotations)
- **26 (41%) removed through banned API removal**

"Security Improvements in Windows Vista", Michael Howard

Google:

- "It's **unreasonable** to expect any developer to be an expert in all these subjects, or to constantly maintain vigilance when writing or reviewing code.
- A better approach is to handle security and reliability in **common frameworks**, **languages**, and **libraries**. Ideally, libraries only expose an interface that makes **writing code with common classes of security vulnerabilities impossible**."

Building Secure and Reliable Systems, by Google



O'REILLY

Building Secure & Reliable Systems

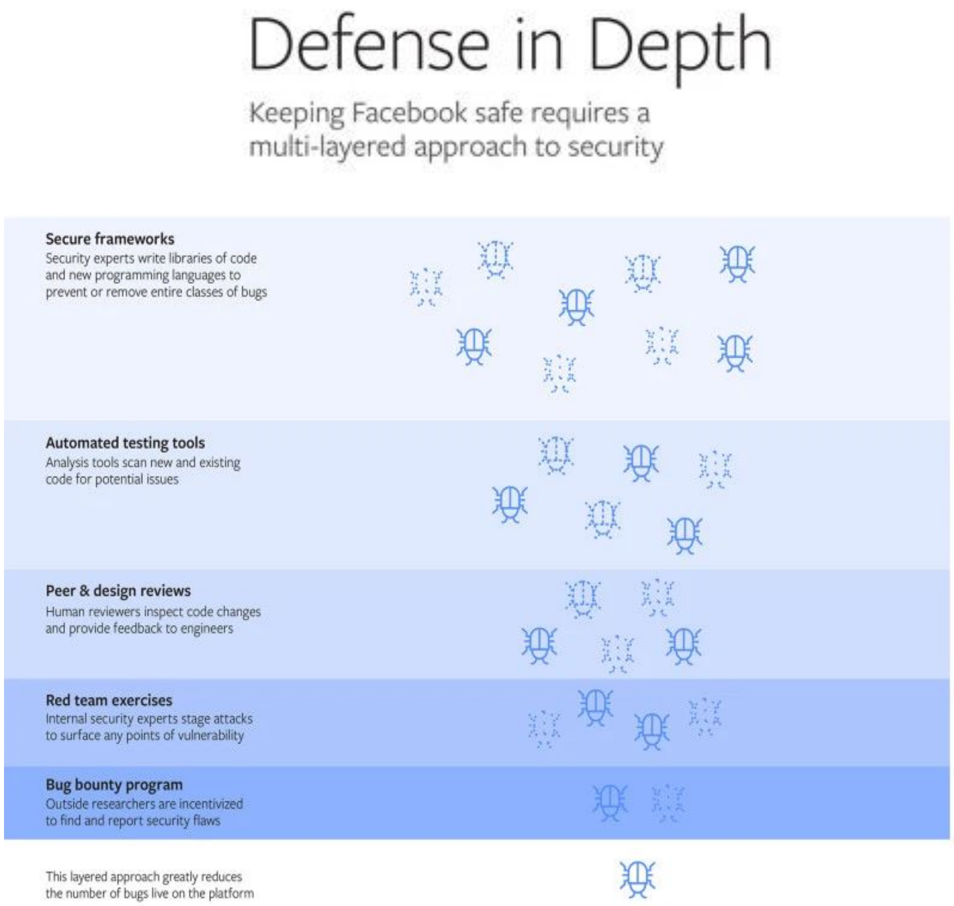
Best Practices for Designing, Implementing and Maintaining Systems

Heather Adkins, Betsy Beyer,
Paul Blankinship, Piotr Lewandowski,
Ana Oprea & Adam Stubblefield

Facebook:

"We invest heavily in building frameworks that help engineers prevent and remove entire classes of bugs when writing code."

Designing Security For Billions by Facebook



Defense in Depth

Keeping Facebook safe requires a multi-layered approach to security

Secure frameworks
Security experts write libraries of code and new programming languages to prevent or remove entire classes of bugs

Automated testing tools
Analysis tools scan new and existing code for potential issues

Peer & design reviews
Human reviewers inspect code changes and provide feedback to engineers

Red team exercises
Internal security experts stage attacks to surface any points of vulnerability

Bug bounty program
Outside researchers are incentivized to find and report security flaws

This layered approach greatly reduces the number of bugs live on the platform

2022 SAN FRANCISCO BSIDES FROM THE GROUND UP

The Power of Guardrails

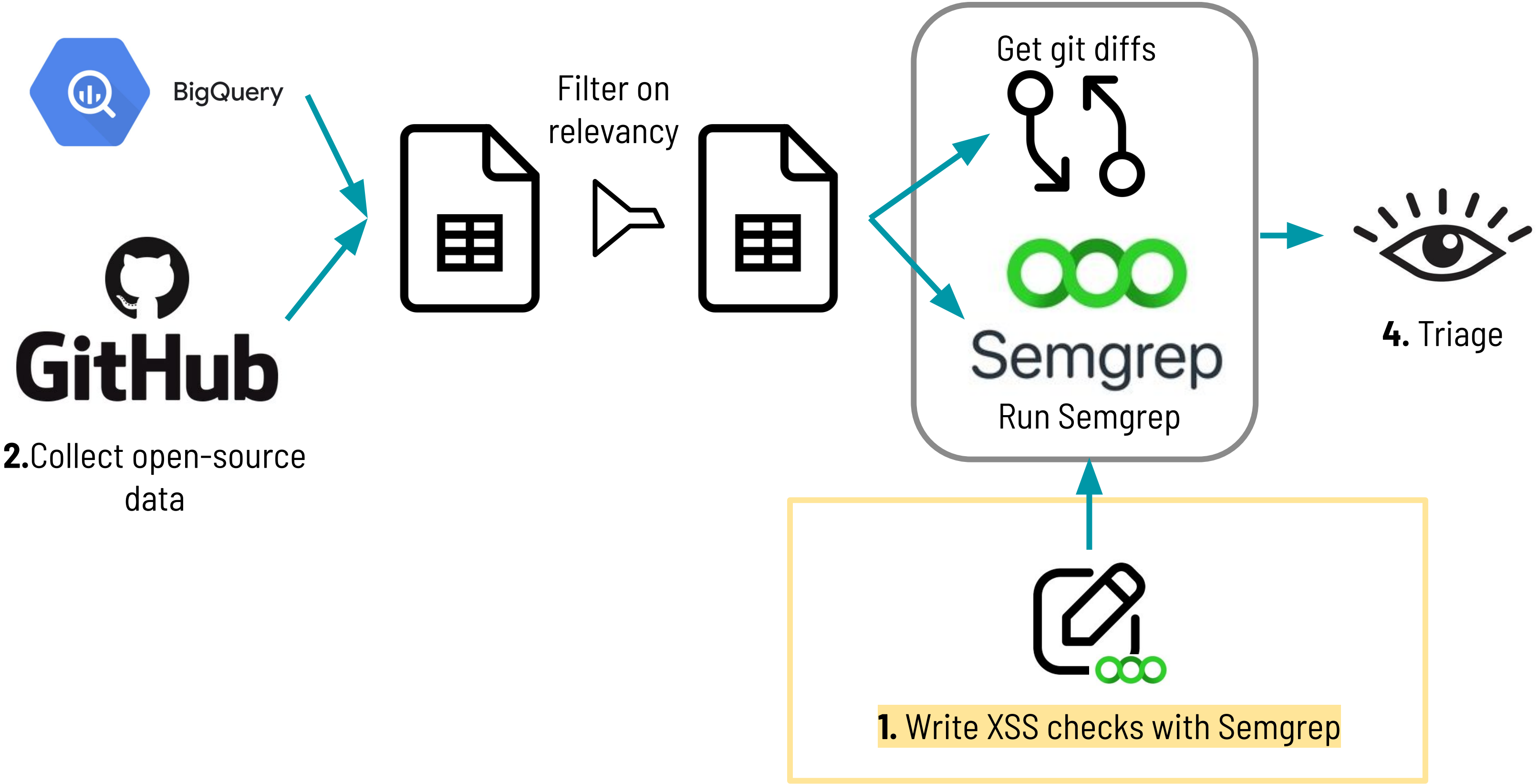
How to Slash Your Risk of XSS in Half

Grayson Hardaway & Colleen Dai

🐦 @r2cdev

Slides: https://bit.ly/2022-BSidesSF-XSS-Guardrails

Architecture



59% could have been prevented

Total number of repositories	125

Total number of distinct commits	140
Total number of detected XSS (true positives)	82
Detection rate	**58.57%**

“But I’m not Google”

Framework / tech choices matter

- Mitigate classes of vulnerabilities

Examples:

- Using modern web frameworks & libraries
- DOMPurify - XSS sanitizer
- re2 - regexes
- tink - crypto
- Write your internal secure XML parser library
- Segment ui-box - safeHref.ts

Web security before modern frameworks & libraries

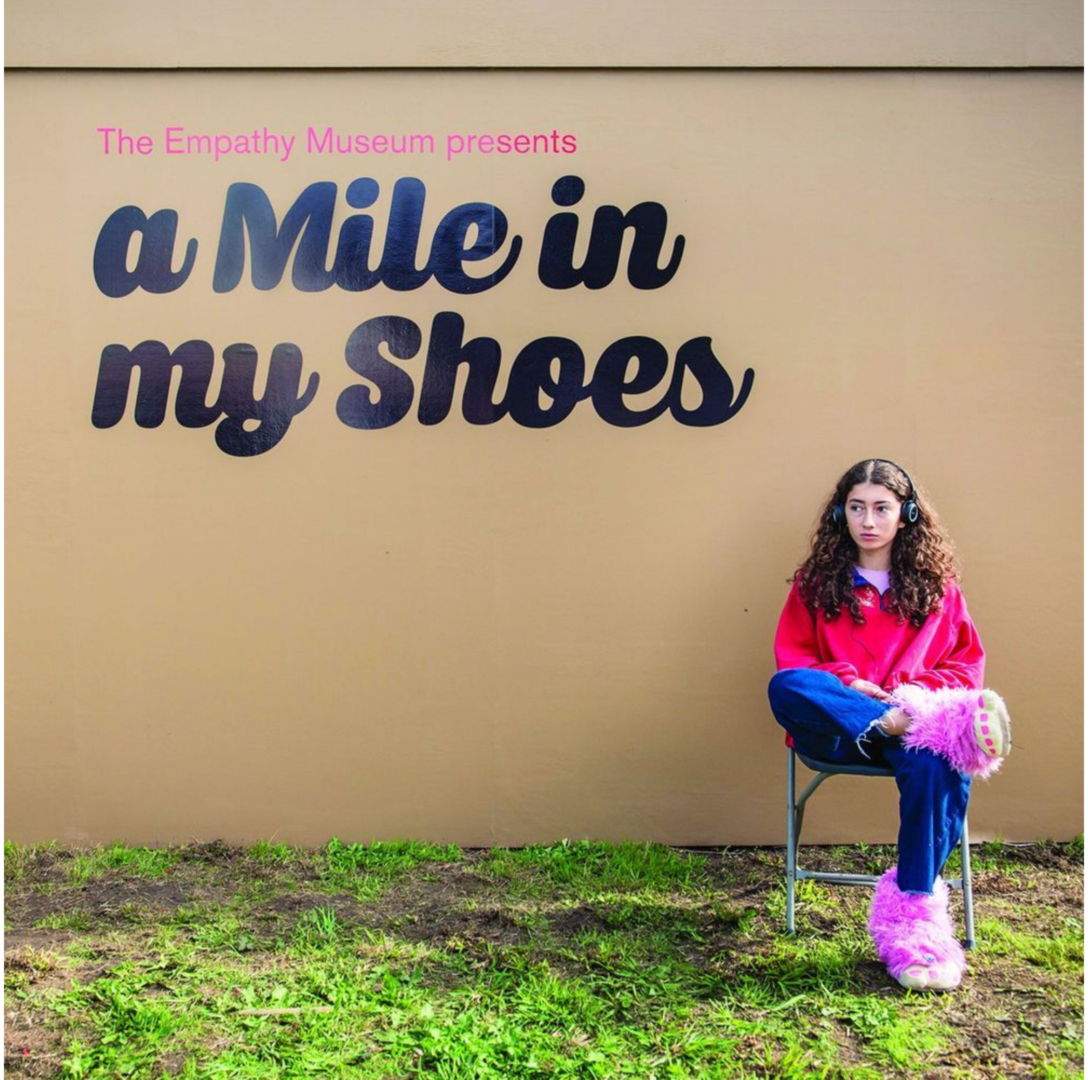
How to Eradicate Vulnerability Classes

1. Evaluate which vulnerability class to focus on
2. Determine the best approach to find/prevent it at scale
3. Select a safe pattern and make it the default
4. Train developers to use the safe pattern
5. Use tools to enforce the safe pattern

Scaling Security, Building Relationships, Build the Future

?

Clint Gibler | @clintgibler | tldrsec.com
R2c.dev | @r2cdev | r2c Community Slack


The Empathy Museum presents
a Mile in my Shoes