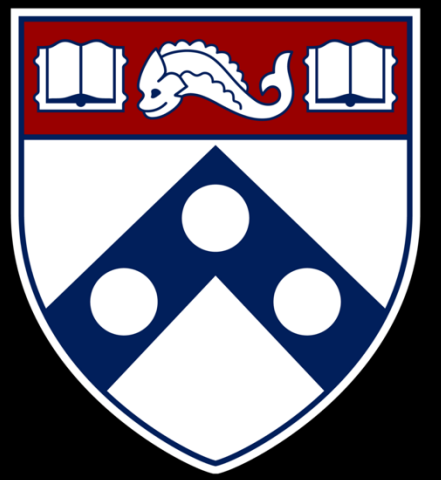
Secure Systems Engineering and Management

A Data-driven Approach



Recap

Michael Hicks

UPenn CIS 7000-003
Spring 2026

Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security
- **Follow a well-designed process for secure systems construction**, from threat modeling to building to testing to maintenance
- **Manage security operations** – preventing, detecting, mitigating, and recovering from incidents – and gather data to improve future posture
- **Make risk-informed decisions**: Assess designs and technologies according to how they mitigate security risk, while leveraging **insurance** and responding to **regulation**
- **Communicate effectively and with empathy** to key stakeholders about security options and recommendations

All while taking a data-informed approach

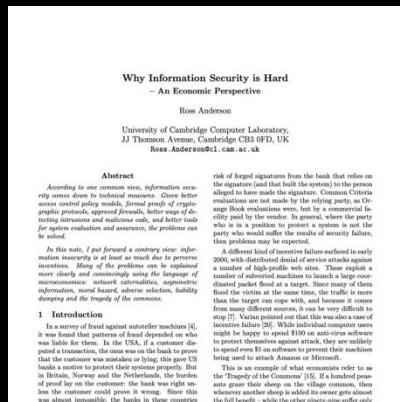
Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security

All while taking a data-informed approach

Why Information Security is Hard
An Economic Perspective

- Party best-placed to improve security should be incentivized to do so
  - Liability
  - Tragedy of the commons
  - First-mover and network externalities vs. security
  - 3rd party evaluators paid by seller vs. direct user evaluation
- Costs of attack vs. defense
  - “Even a very moderately resourced attacker can break anything that’s at all large and complex”
- Market forces: end-user security vs. developer pain
- “In an ideal world, the removal of perverse economic incentives to create insecure systems would depoliticize most issues.”



Why Information Security is Hard
– An Economic Perspective

Ross Anderson

University of Cambridge Computer Laboratory,
JJ Thomson Avenue, Cambridge CB3 0FD, UK
Ross.Anderson@cl.cam.ac.uk

Abstract

According to one common view, information security comes down to technical measures. Given better access control policy models, formal proofs of cryptographic protocols, approved firewalls, better ways of detecting intrusions and malicious code, and better tools for system evaluation and assurance, the problems can be solved.

In this note, I put forward a contrary view: information insecurity is at least as much due to perverse incentives. Many of the problems can be explained more clearly and convincingly using the language of microeconomics: network externalities, asymmetric information, moral hazard, adverse selection, liability dumping and the tragedy of the commons.

1 Introduction

In a survey of fraud against autoteller machines [4], it was found that patterns of fraud depended on who was liable for them. In the USA, if a customer disputed a transaction, the onus was on the bank to prove that the customer was mistaken or lying; this gave US banks a motive to protect their systems properly. But in Britain, Norway and the Netherlands, the burden of proof lay on the customer: the bank was right unless the customer could prove it wrong. Since this was almost impossible, the banks in these countries risk of forged signatures from the bank that relies on the signature (and that built the system) to the person alleged to have made the signature. Common Criteria evaluations are not made by the relying party, as Orange Book evaluations were, but by a commercial facility paid by the vendor. In general, where the party who is in a position to protect a system is not the party who would suffer the results of security failure, then problems may be expected.

A different kind of incentive failure surfaced in early 2000, with distributed denial of service attacks against a number of high-profile web sites. These exploit a number of subverted machines to launch a large coordinated packet flood at a target. Since many of them flood the victim at the same time, the traffic is more than the target can cope with, and because it comes from many different sources, it can be very difficult to stop [7]. Varian pointed out that this was also a case of incentive failure [20]. While individual computer users might be happy to spend $100 on anti-virus software to protect themselves against attack, they are unlikely to spend even $1 on software to prevent their machines being used to attack Amazon or Microsoft.

This is an example of what economists refer to as the ‘Tragedy of the Commons’ [15]. If a hundred peasants graze their sheep on the village common, then whenever another sheep is added its owner gets almost

Silver Bullets, not Lemons

Extends Anderson:

- Security is a good
- Security is hard to assess
  - Leads to information insufficiency *on both sides*


The Market for Silver Bullets

iang.org/papers/market_for_silver_bullets.html

The Market for Silver Bullets

Ian Grigg
Systemics, Inc.

2nd March 2008

Abstract: What is security?

As a "good" in the sense of economics, security is now recognised as being one for which our knowledge is poor. As with safety goods, events of utility tend to be destructive, yet unlike safety goods, the performance of the good is very hard to test. The roles of participants are complicated by the inclusion of agressive attackers, and buyers and sellers that interchange.

This essay hypothesises that security is a good with insufficient information, and rejects the assumption that security fits in the market for goods with asymmetric information. Security can be viewed as a market where neither buyer nor seller has sufficient information to be able to make a rational buying decision. Drawing heavily from Michael Spence's "Job Market Signaling," these characteristics lead to the arisal of a market in *silver bullets* as participants *herd* in search of *best practices*, a common set of goods that arises more to reduce the costs of externalities rather than achieve benefits in security itself.

The Market for Goods, as described by Information and by Party	**Buyer Knows**	**Buyer Lacks *H1***
Seller Knows	Efficient Goods	Lemons (used cars)
Seller Lacks *H2*	Limes (Insurance)	**Silver Bullets (Security)**

Lacks empirical basis,
i.e., *falsifiability*

A Fundamental Asymmetry

We can observe:

✅ Insecurity (attacks succeed, systems fail)

We cannot observe:

❌ Security (absence of attacks ≠ security)

This is a **fundamental** asymmetry.

SoK: Science, Security, and the Elusive Goal of Security as a Scientific Pursuit

Cormac Herley
Microsoft Research, Redmond, WA, USA
cormac@microsoft.com

P.C. van Oorschot
Carleton University, Ottawa, ON, Canada
paulv@scs.carleton.ca

***Abstract*—The past ten years has seen increasing calls to make security research more "scientific". On the surface, most agree that this is desirable, given universal recognition of "science" as a positive force. However, we find that there is little clarity on what "scientific" means in the context of computer security research, or consensus on what a "Science of Security" should look like. We selectively review work in the history and philosophy of science and more recent work under the label "Science of Security". We explore what has been done under the theme of relating science and security, put this in context with historical science, and offer observations and insights we hope may motivate further exploration and guidance. Among our findings are that practices on which the rest of science has reached consensus appear little used or recognized in security, and a pattern of methodological errors continues unaddressed.**

***Index Terms*—security research; science of security; history of science; philosophy of science; connections between research and observable world.**

I. Introduction and Overview

Security is often said to have unique challenges. Progress can be harder to measure than in areas where, e.g., performance metrics or capabilities point to visible steady improvement. Supposedly unique factors, such as the presence of active adversaries, complicate matters. Some even describe the field in pessimistic terms. Multics warriors remind the young that many of today's problems were much better addressed forty years ago [1]. Shamir, in accepting the 2002 Turing award, described non-crypto security as "a mess." Schell, in 2001, described the field as being filled with "pseudo-science and flying pigs" [2].

Perhaps in response to these negative views, over the last decade there has been an effort in parts of the community to develop a "Science of Security" (SoS). In this paper we review both work in the history/philosophy of science and, recently, under this SoS banner. We wish to distinguish at the outset between these two strands. The first is an exploration of the techniques that the consensus from other fields suggest are important to pursuing any problem scientifically. The second is the activity and body of work that has resulted from external promotion of an agenda by the name "Science of Security".

research) in the light of consensus views of science and scientific methods. We find that aspects from the philosophy of science on which most other communities have reached consensus appear surprisingly little used in security, including in work done under the SoS label. For example, we do not find that that work better adheres to scientific principles than other security research in any readily identifiable way.

We identify several opportunities that may help drive security research forward in a more scientific fashion, and on this we are cautiously optimistic. While we see great benefit to this, we also do not wish to argue that all of security must be done on rigidly scientific principles. A significant component of security is engineering; this shares with science the regular contact with, and feedback from, observation, despite not having as clearly articulated a definition or methods.

Section II selectively reviews literature on the history and philosophy of science, with particular emphasis on three things: 1) methodologies and positions on which practicing scientists and philosophers of science have largely reached consensus; 2) aspects highlighting opportunities to eliminate confusion in security research; and 3) contributions pointing to where security research might be made "more scientific". Section III selectively reviews literature relating "science" and "security", for examples of viewpoints within the community, for context in later discussion, and as supporting evidence for arguments; an exhaustive review of all security literature attempting to determine which papers use scientific methods in security research is not a goal. Section IV highlights areas where the security community has failed to adopt accepted lessons from the science literature. Section V provides insights and offers observations and constructive suggestions. Section VI concludes.

II. History/Philosophy of Science

This section highlights aspects from the history and philosophy of science most relevant to security research. Our goal here is not an encyclopedic review of science literature;

Deduction vs. Induction

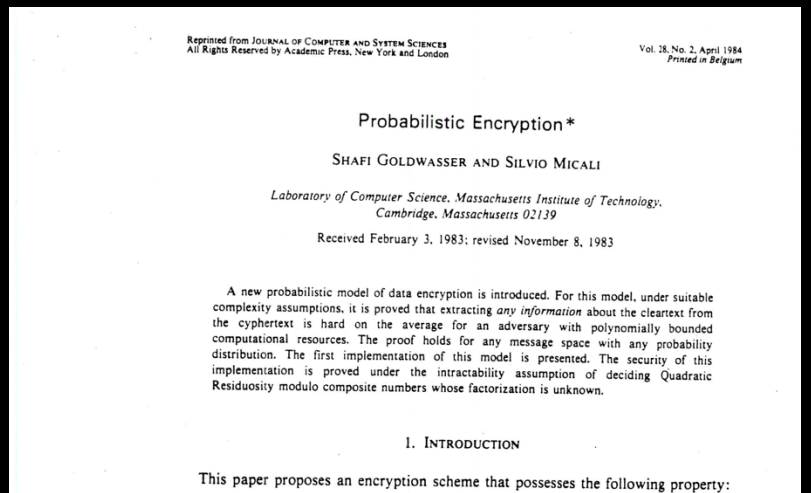
“Deduction in itself is quite powerful as a method of scientific discovery…”

- With a good model, you can deduce surprising and useful results
  - Shannon's channel capacity theorem (1948)
  - Gödel's incompleteness theorems (1931)
  - Deductions from Euclidean geometry (e.g., Pythagorean theorem)

Key challenge: Specifying security

To know if a system is secure, we have to

- Develop a model for it
- Describe what the model should (and should not) do
- Prove the model satisfies our description
- Establish that the model, and its assumptions, represent reality



Reprinted from JOURNAL OF COMPUTER AND SYSTEM SCIENCES
All Rights Reserved by Academic Press, New York and London
Vol. 28, No. 2, April 1984
Printed in Belgium

Probabilistic Encryption*

SHAFI GOLDWASSER AND SILVIO MICALI

Laboratory of Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Received February 3, 1983; revised November 8, 1983

A new probabilistic model of data encryption is introduced. For this model, under suitable complexity assumptions, it is proved that extracting *any information* about the cleartext from the cyphertext is hard on the average for an adversary with polynomially bounded computational resources. The proof holds for any message space with any probability distribution. The first implementation of this model is presented. The security of this implementation is proved under the intractability assumption of deciding Quadratic Residuosity modulo composite numbers whose factorization is unknown.

1. INTRODUCTION

This paper proposes an encryption scheme that possesses the following property:

Formally Verified Cloud-Scale Authorization

Aleks Chakarov *Amazon* aleksach@
Jaco Geldenhuys *Amazon* jgeldenh@
Matthew Heck *Amazon* mcheck@
Michael Hicks *Amazon* mwhicks@
Sam Huang *Amazon* srhuang@
Georges-Axel Jaloyan *Amazon* gjaloyan@
Anjali Joshi *Amazon* anjalijs@
K. Rustan M. Leino *Amazon* leino@
Mikael Mayer *Amazon* mimayere@
Sean McLaughlin *Amazon* seanmcl@
Akhilesh Mritunjai *Amazon* amritun@
Clement Pit-Claudel *Amazon* pitclauc@amazon.ch
Sorawee Porncharoenwase *Amazon* soraweep@
Florian Rabe *Amazon* florrabe@
Marianna Rapoport *Amazon* rapopor@
Giles Reger *Amazon* reggiles@
Cody Roux *Amazon* codyroux@
Neha Rungta *Amazon* rungta@
Robin Salkeld *Amazon* salkeldr@
Matthias Schlaipfer *Amazon* schlaipf@
Daniel Schoepe *Amazon* schoeped@
Johanna Schwartzentruber *Amazon* jjsch@
Serdar Tasiran *Amazon* tasirans@
Aaron Tomb *Amazon* aarotomb@
Emina Torlak *Amazon* torlaket@
Jean-Baptiste Tristan *Amazon* trjohnb@
Lucas Wagner *Amazon* lgwagner@
Michael W. Whalen *Amazon* mww@
Remy Willems *Amazon* rwillems@
Tongtong Xiang *Amazon* ttx@
Tae Joon Byun *Meta*
Joshua Cohen *Princeton University*
Ruijie Fang *University of Texas at Austin*
Junyoung Jang *McGill University*

Fundamentally an empirical question

Evidence-based security

Analysis

- What are the (still) successful vectors of attack?
- Where is risk (still) greatest?
- What interventions could be deployed cost-effectively?

Outcomes

- Hosts compromised,
- Vulnerabilities exploited,
- Revenue lost, …

Intervention

- Engineering,
- Operations,
- Policy,
- Education, …

What to measure?

Outcomes

- Hosts compromised,
- Vulnerabilities exploited,
- Revenue lost, …

Intervention

- Engineering,
- Operations,
- Policy,
- Education, …

Measuring Security Practices and How They Impact Security

Louis F. DeKoven
University of California, San Diego
ldekoven@cs.ucsd.edu

Audrey Randall
University of California, San Diego
aurandal@eng.ucsd.edu

Ariana Mirian
University of California, San Diego
amirian@cs.ucsd.edu

Gautam Akiwate
University of California, San Diego
gakiwate@cs.ucsd.edu

Ansel Blume
University of California, San Diego
ablume@ucsd.edu

Lawrence K. Saul
University of California, San Diego
saul@cs.ucsd.edu

Aaron Schulman
University of California, San Diego
schulman@cs.ucsd.edu

Geoffrey M. Voelker
University of California, San Diego
voelker@cs.ucsd.edu

Stefan Savage
University of California, San Diego
savage@cs.ucsd.edu

ABSTRACT

Security is a discipline that places significant expectations on lay users. Thus, there are a wide array of technologies and behaviors that we exhort end users to adopt and thereby reduce their security risk. However, the adoption of these "best practices" — ranging from the use of antivirus products to actively keeping software updated — is not well understood, nor is their practical impact on security risk well-established. This paper explores both of these issues via a large-scale empirical measurement study covering approximately 15,000 computers over six months. We use passive monitoring to infer and characterize the prevalence of various security practices in situ as well as a range of other potentially security-relevant behaviors. We then explore the extent to which differences in key security behaviors impact real-world outcomes (i.e., that a device shows clear evidence of having been compromised).

CCS CONCEPTS

• **Security and privacy** → *Intrusion detection systems*; • **Networks**;

ACM Reference Format:
Louis F. DeKoven, Audrey Randall, Ariana Mirian, Gautam Akiwate, Ansel Blume, Lawrence K. Saul, Aaron Schulman, Geoffrey M. Voelker, and Stefan Savage. 2019. Measuring Security Practices and How They Impact Security. In *Internet Measurement Conference (IMC '19), October 21–23, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3355369.3355571

1 INTRODUCTION

Ensuring effective computer security is widely understood to require a combination of both appropriate technological measures and prudent human behaviors; e.g., rapid installation of security updates to patch vulnerabilities or the use of password managers to ensure login credentials are distinct and random. Implicit in this status quo is the recognition that security is not an intrinsic property of today's systems, but is a byproduct of making appropriate choices — choices about what security products to employ, choices about how to manage system software, and choices about how to engage (or not) with third-party services on the Internet. Indeed, the codifying of good security choices, commonly referred to as security policy or "best practice", has been a part of our lives as long as security has been a concern.

However, establishing the value provided by these security practices is underexamined at best. First, we have limited empirical data about which security advice is adopted in practice. Users have a plethora of advice to choose from, highlighted by Reeder et al.'s recent study of expert security advice, whose title — "152 Simple Steps to Stay Safe Online" — underscores both the irony and the variability in such security lore [35]. Clearly few users are likely to follow all such dicta, but if user behavior is indeed key to security, it is important to know which practices are widely followed and which have only limited uptake.

A second, more subtle issue concerns the efficacy of security practices when followed: Do they work? Here the evidence is scant. Even practices widely agreed upon by Reeder's experts, such as keeping software patched, are not justified beyond a rhetorical argument. In fact, virtually all of the most established security best practices — including "use antivirus software", "use HTTPS/TLS", "update your software regularly", "use a password manager", and so on — have attained this status without empirical evidence quantifying their impact on security outcomes. Summarizing this state of affairs, Herley writes, "[Security] advice is complex and growing, but the benefit is largely speculative or moot", which he argues leads rational users to reject security advice [17].

To summarize, our existing models of security all rely on end users to follow a range of best practices. However, we neither understand the extent to which they are following this advice, nor do we have good information about how much this behavior ultimately impacts their future security.

This paper seeks to make progress on both issues — the prevalence of popular security practices and their relationship to security outcomes — via longitudinal empirical measurement of a large population of computer devices. In particular, we monitor the online

IMC '19, October 21–23, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-6948-0/19/10…$15.00
https://doi.org/10.1145/3355369.3355571

What to measure?

Outcomes

- Hosts compromised,
- Vulnerabilities exploited,
- Revenue lost, …

Intervention

- Engineering,
- Operations,
- Policy,
- Education, …

A Large-Scale Measurement of Cybercrime Against Individuals

Casey F. Breen
caseybreen@berkeley.edu
University of California, Berkeley
Berkeley, CA, USA

Cormac Herley
cormac@microsoft.com
Microsoft Research
Redmond, WA, USA

Elissa M. Redmiles
eredmiles@mpi-sws.org
Max Planck Institute
for Software Systems
Saarbrucken, Germany

ABSTRACT

We know surprisingly little about the prevalence and severity of cybercrime in the U.S. Yet, in order to prioritize the development and distribution of advice and technology to protect end users, we require empirical evidence regarding cybercrime. Measuring crime, including cybercrime, is a challenging problem that relies on a combination of direct crime reports to the government – which have known issues of under-reporting – and assessment via carefully-designed self-report surveys. We report on the first large-scale, nationally representative academic survey (n=11,953) of consumer cybercrime experiences in the U.S. Our analysis answers four research questions: (1) What is the prevalence and (2) the monetary impact of these cybercrimes we measure in the U.S.?, (3) Do inequities exist in victimization?, and (4) Can we improve cybercrime measurement by leveraging social-reporting techniques used to measure physical crime? Our analysis also offers insight toward improving future measurement of cybercrime and protecting users.

CCS CONCEPTS

• **Human-centered computing** → **Empirical studies in HCI**; *User studies*; • **Security and privacy** → **Economics of security and privacy**.

KEYWORDS

cybercrime, network scale-up, digitial inequity

ACM Reference Format:
Casey F. Breen, Cormac Herley, and Elissa M. Redmiles. 2022. A Large-Scale Measurement of Cybercrime Against Individuals. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 41 pages. https://doi.org/10.1145/3491102.3517613

1 INTRODUCTION

While cybercrime protection is an area of significant focus in human-computer interaction (HCI) research [10], relatively little is known about the prevalence and severity of the cybercrimes we aim to prevent. Most efforts to quantify the size and cost of crime still focus solely on physical crimes (e.g., robbery, assault), ignoring the reality of digital victimization [5, 6].

CHI '22, April 29-May 5, 2022, New Orleans, LA, USA

ACM ISBN 978-1-4503-9157-3/22/04.
https://doi.org/10.1145/3491102.3517613

Yet, in an empirical science of HCI, "measurements create certain possibilities for action and exclude other possibilities" [59]. That is, data – or a lack of it – guides system design. In the presence of data on people's digital experiences of crime (i.e., cybercrime incidence), for example, HCI researchers and security technologists may prioritize the design of certain cybercrime protections over others. In the absence of such data, researchers may instead privilege the goals of technology companies or state entities that fund their research [59] or turn to computational transformations – "solve[ing] a computationally tractable transformation of a problem rather than the problem itself" [7] – to prioritize design and intervention. As such, recent research in HCI [10] and in cybersecurity [66] calls for measurement of cybercrime to provide appropriate context for the data-driven design of interventions, with the former noting that: "it is critical that we examine and make explicit the impact of crime...to inform safer, intelligent and just digital and non-digital spaces for all."

No prior academic work has focused on survey-based measurements of cybercrime incidence in the U.S., nor has prior academic work, within or outside of the U.S., investigated potential inequities in the prevalence of these crimes (see Figure 1). The latter investigation is critical to ensure that we address these crimes equitably across user groups. In this study, we take a first step toward filling this measurement gap by conducting a probabilistic, nationally-representative survey of 11,953 Americans to measure the prevalence of six exemplar cybercrimes against individuals in the U.S.: bank account or credit card compromise, non-delivery, non-payment, overpayment, advanced fee scams[1], and digital blackmail / extortion.[2]

Perceived monetary losses are a significant driver of research agendas. For example, research efforts to get users to choose strong passwords or adopt two-factor authentication generally assume that these measures would significantly reduce losses [29]. Work appearing in CHI that addresses efficacy of phishing countermeasures and training [21, 51, 79] routinely cites the Gartner estimates of phishing monetary losses as a justification for research on phishing prevention [1]. As monetary loss is not only a common justification for the prioritization of cybercrime interventions but also the metric used in existing government statistics that are leveraged to decide the funding awarded for cybercrime research, we focus on cybercrimes – computer- or internet-enabled crimes – where a victim suffers a monetary loss.

Our work addresses four primary research questions, the first three of which are:

RQ1 What is the prevalence of six representative cybercrimes in the U.S.?

[1]Best known as "Nigerian Prince" or 419 scams [25].
[2]For more detail regarding our selection criteria see Section 3.1.

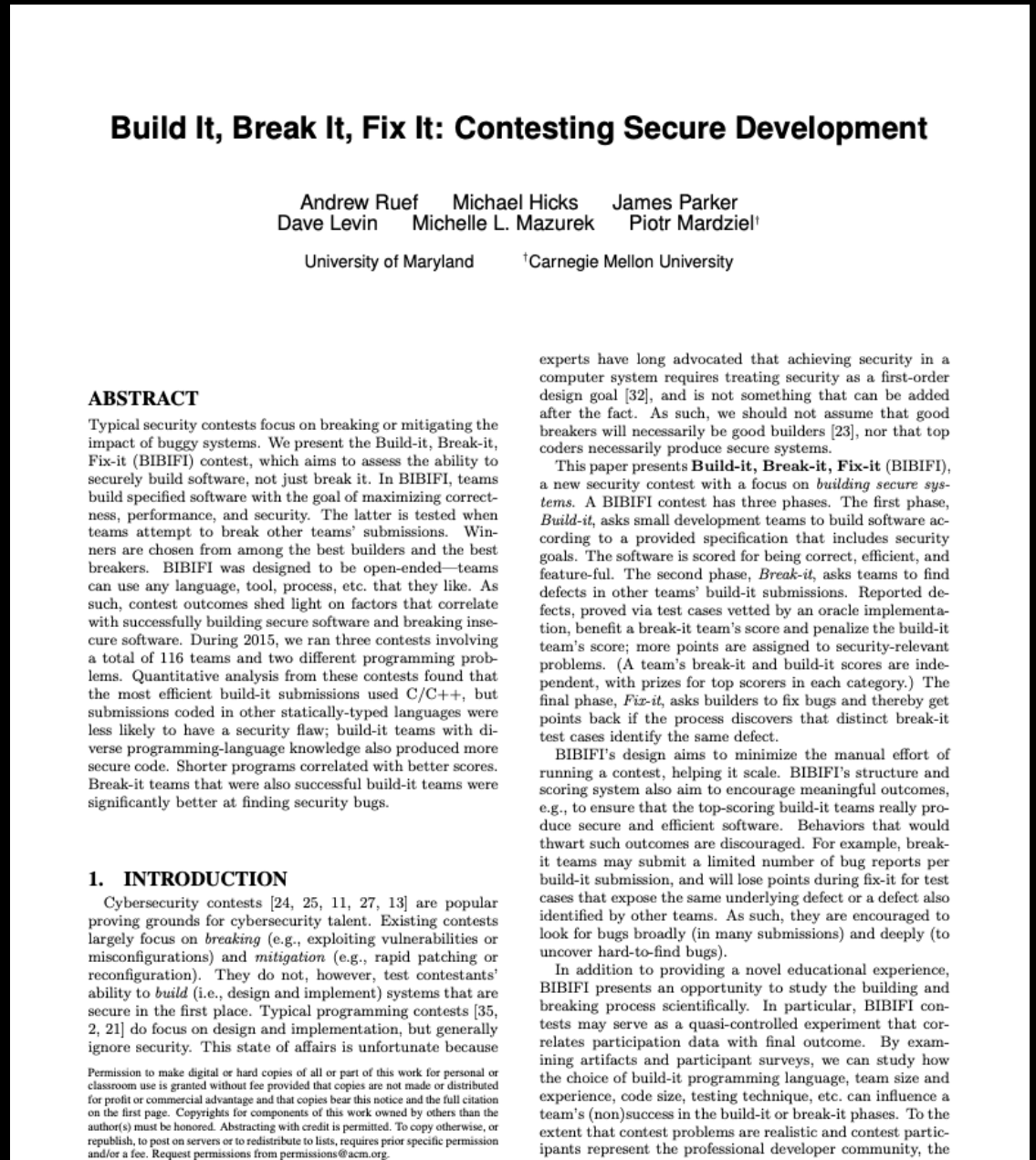
What to measure?

Outcomes

- Hosts compromised,
- Vulnerabilities exploited,
- Revenue lost, …

Intervention

- Engineering,
- Operations,
- Policy,
- Education, …



Build It, Break It, Fix It: Contesting Secure Development

Andrew Ruef Michael Hicks James Parker
Dave Levin Michelle L. Mazurek Piotr Mardziel†

University of Maryland †Carnegie Mellon University

ABSTRACT

Typical security contests focus on breaking or mitigating the impact of buggy systems. We present the Build-it, Break-it, Fix-it (BIBIFI) contest, which aims to assess the ability to securely build software, not just break it. In BIBIFI, teams build specified software with the goal of maximizing correctness, performance, and security. The latter is tested when teams attempt to break other teams' submissions. Winners are chosen from among the best builders and the best breakers. BIBIFI was designed to be open-ended—teams can use any language, tool, process, etc. that they like. As such, contest outcomes shed light on factors that correlate with successfully building secure software and breaking insecure software. During 2015, we ran three contests involving a total of 116 teams and two different programming problems. Quantitative analysis from these contests found that the most efficient build-it submissions used C/C++, but submissions coded in other statically-typed languages were less likely to have a security flaw; build-it teams with diverse programming-language knowledge also produced more secure code. Shorter programs correlated with better scores. Break-it teams that were also successful build-it teams were significantly better at finding security bugs.

1. INTRODUCTION

Cybersecurity contests [24, 25, 11, 27, 13] are popular proving grounds for cybersecurity talent. Existing contests largely focus on *breaking* (e.g., exploiting vulnerabilities or misconfigurations) and *mitigation* (e.g., rapid patching or reconfiguration). They do not, however, test contestants' ability to *build* (i.e., design and implement) systems that are secure in the first place. Typical programming contests [35, 2, 21] do focus on design and implementation, but generally ignore security. This state of affairs is unfortunate because experts have long advocated that achieving security in a computer system requires treating security as a first-order design goal [32], and is not something that can be added after the fact. As such, we should not assume that good breakers will necessarily be good builders [23], nor that top coders necessarily produce secure systems.

This paper presents **Build-it, Break-it, Fix-it** (BIBIFI), a new security contest with a focus on *building secure systems*. A BIBIFI contest has three phases. The first phase, *Build-it*, asks small development teams to build software according to a provided specification that includes security goals. The software is scored for being correct, efficient, and feature-ful. The second phase, *Break-it*, asks teams to find defects in other teams' build-it submissions. Reported defects, proved via test cases vetted by an oracle implementation, benefit a break-it team's score and penalize the build-it team's score; more points are assigned to security-relevant problems. (A team's break-it and build-it scores are independent, with prizes for top scorers in each category.) The final phase, *Fix-it*, asks builders to fix bugs and thereby get points back if the process discovers that distinct break-it test cases identify the same defect.

BIBIFI's design aims to minimize the manual effort of running a contest, helping it scale. BIBIFI's structure and scoring system also aim to encourage meaningful outcomes, e.g., to ensure that the top-scoring build-it teams really produce secure and efficient software. Behaviors that would thwart such outcomes are discouraged. For example, break-it teams may submit a limited number of bug reports per build-it submission, and will lose points during fix-it for test cases that expose the same underlying defect or a defect also identified by other teams. As such, they are encouraged to look for bugs broadly (in many submissions) and deeply (to uncover hard-to-find bugs).

In addition to providing a novel educational experience, BIBIFI presents an opportunity to study the building and breaking process scientifically. In particular, BIBIFI contests may serve as a quasi-controlled experiment that correlates participation data with final outcome. By examining artifacts and participant surveys, we can study how the choice of build-it programming language, team size and experience, code size, testing technique, etc. can influence a team's (non)success in the build-it or break-it phases. To the extent that contest problems are realistic and contest participants represent the professional developer community, the

How to analyze your data?

- Hypothesis testing
- Chi-square test of independence
- Student’s t-test (parametric)
- Mann-Whitney U test (non-parametric)
- Effect sizes (Cohen’s d, Vargha-Delaney A)
- Bootstrapped confidence intervals
- Linear regression models
- Categorical predictors (dummy coding)
- Model comparison (likelihood-ratio tests)
- Logistic regression for binary outcomes
- Odds ratios and predicted probabilities

How to analyze your data? Beware!

Misuse, Misreporting, Misinterpretation of Statistical Methods in Usable Privacy and Security Papers

Jenny Tang
Carnegie Mellon University

Lujo Bauer
Carnegie Mellon University

Nicolas Christin
Carnegie Mellon University

Abstract

Null hypothesis significance testing (NHST) is commonly used in quantitative usable privacy and security studies. Many papers use results from statistical tests to assert whether effects or differences exist depending on the resulting p-value. We conduct a systematic review of papers published in 10 editions of the Symposium on Usable Privacy and Security over a span of 20 years to evaluate the field's use of NHST. We code statistical tests for potential statistical validity, reporting, or interpretation issues that may undermine assertions made in the 121 papers that use NHST. Most problematically, tests in 23% of papers inadequately account for non-independence between samples, leading to potentially invalid claims. 58% of papers lack information to verify whether an assertion is supported, such as imprecisely specifying the statistical test conducted. Many papers contain more minor statistical issues or report statistics in ways that deviate from best practice. We conclude with recommendations for statistical reporting and statistical thinking in the field.

1 Introduction

Statistical methods are often used in human-computer interaction research to support assertions about the presence (or absence) of an effect of scientific significance (e.g., some magnitude of difference) accompanied by a measure of statistic significance. Indeed, one of the most common refrains in statistical analysis is that a result is significant because the "p-value" is less than a given threshold, e.g., $p < 0.05$. Despite over half a century of criticism, null hypothesis significance testing (NHST, also known as statistical significance testing)—that is, methods using p-values from inferential statistical tests as evidence to reject a null hypothesis—remains the dominant form of statistical analysis and evaluation [17]. However, simply dichotomizing results into "significant" and "non-significant" through their associated p-values without reporting other information is not in itself sufficient to convey the scientific importance of the claims, nor the richness and complexity of data collected from human subjects. This reliance on p-values to support assertions sometimes leads other information vital to understanding statistical and scientific significance to be omitted.

As a result, complete reliance on p-values is increasingly frowned upon, with some journals banning the reporting of p-values altogether [75, 81]. Most other current guidance is less drastic, and recommends using statistical hypothesis testing as a starting point and providing sufficient context (such as effect sizes, confidence intervals, and underlying data) to convey the scientific significance of the claims [2,13,49,59, 80,81]. We use this guidance to evaluate whether the scientific assertions made on the basis of NHST in usable privacy and security (UPS) are accompanied by sufficient reporting for readers to validate whether these assertions are supported by the information present in the paper. We focus on UPS as it is still a fairly young area, with evolving standards, features a considerable amount of quantitative research, and errors or misinterpretations can be detrimental to user safety in the digital world and beyond.

Prior work has also examined the transparency, reporting, and validity of statistical methods in HCI and various subfields [16, 25, 36, 51, 62, 66, 77]. However, the evaluations in these works typically focus on evaluating whether p-values are accurately computed or on whether there may be false negatives (such as due to lack of power) or false positives (such as from inaccurately reported p-values).

In this work, we look beyond statistical significance to examine statistical validity (whether the chosen test is suitable for the data or whether it may produce spurious results), reporting transparency and completeness (whether the reported

USENIX Symposium on Usable Privacy and Security (SOUPS) 2025.
August 10–12, 2025, Seattle, WA, United States.

Evaluating Fuzz Testing

George Klees, Andrew Ruef, Benji Cooper
University of Maryland

Shiyi Wei
University of Texas at Dallas

Michael Hicks
University of Maryland

ABSTRACT

Fuzz testing has enjoyed great success at discovering security critical bugs in real software. Recently, researchers have devoted significant effort to devising new fuzzing techniques, strategies, and algorithms. Such new ideas are primarily evaluated experimentally so an important question is: What experimental setup is needed to produce trustworthy results? We surveyed the recent research literature and assessed the experimental evaluations carried out by 32 fuzzing papers. We found problems in every evaluation we considered. We then performed our own extensive experimental evaluation using an existing fuzzer. Our results showed that the general problems we found in existing experimental evaluations can indeed translate to actual wrong or misleading assessments. We conclude with some guidelines that we hope will help improve experimental evaluations of fuzz testing algorithms, making reported results more robust.

CCS CONCEPTS

• **Security and privacy → Software and application security**;

KEYWORDS

fuzzing, evaluation, security

ACM Reference Format:
George Klees, Andrew Ruef, Benji Cooper, Shiyi Wei, and Michael Hicks. 2018. Evaluating Fuzz Testing. In *2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), October 15–19, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3243734.3243804

1 INTRODUCTION

A *fuzz tester* (or *fuzzer*) is a tool that iteratively and randomly generates inputs with which it tests a target program. Despite appearing "naive" when compared to more sophisticated tools involving SMT solvers, symbolic execution, and static analysis, fuzzers are surprisingly effective. For example, the popular fuzzer AFL has been used to find hundreds of bugs in popular programs [1]. Comparing AFL head-to-head with the symbolic executor *angr*, AFL found 76% more bugs (68 vs. 16) in the same corpus over a 24-hour period [50]. The success of fuzzers has made them a popular topic of research.

Why do we think fuzzers work? While inspiration for new ideas may be drawn from mathematical analysis, fuzzers are primarily evaluated experimentally. When a researcher develops a new fuzzer algorithm (call it A), they must empirically demonstrate that it provides an advantage over the status quo. To do this, they must choose:

- a compelling *baseline* fuzzer B to compare against;
- a sample of target programs—the *benchmark suite*;
- a *performance metric* to measure when A and B are run on the benchmark suite; ideally, this is the number of (possibly exploitable) bugs identified by crashing inputs;
- a meaningful set of *configuration parameters*, e.g., the *seed file* (or files) to start fuzzing with, and the *timeout* (i.e., the duration) of a fuzzing run.

An evaluation should also account for the fundamentally random nature of fuzzing: Each fuzzing run on a target program may produce different results than the last due to the use of randomness. As such, an evaluation should measure *sufficiently many trials* to sample the overall distribution that represents the fuzzer's performance, using a *statistical test* [38] to determine that A's measured improvement over B is real, rather than due to chance.

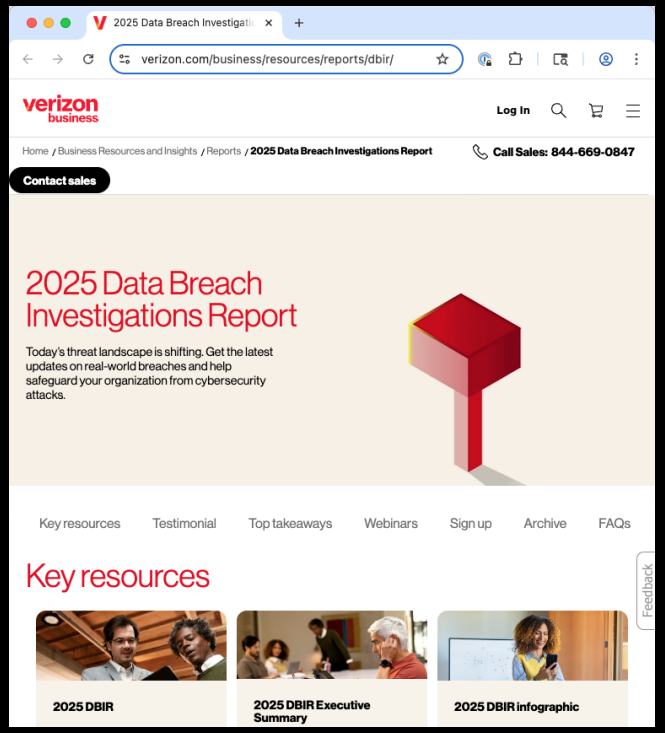
Failure to perform one of these steps, or failing to follow recommended practice when carrying it out, could lead to misleading or incorrect conclusions. Such conclusions waste time for practitioners, who might profit more from using alternative methods or configurations. They also waste the time of researchers, who make overly strong assumptions based on an arbitrary tuning of evaluation parameters.

We examined 32 recently published papers on fuzz testing (see Table 1) located by perusing top-conference proceedings and other quality venues, and studied their experimental evaluations. We found that no fuzz testing evaluation carries out all of the above steps properly (though some get close). This is bad news in theory, and after carrying out more than 50000 CPU hours of experiments, we believe it is bad news in practice, too. Using AFLFast [6] (as A) and AFL (as baseline B), we carried out a variety of tests of their performance. We chose AFLFast as it was a recent advance over the state of the art; its code was publicly available; and we were confident in our ability to rerun the experiments described by the authors in their own evaluation and expand these experiments by varying parameters that the original experimenters did not. This choice was also driven by the importance of AFL in the literature: 14 out of 32 papers we examined used AFL as a baseline in their evaluation. We targeted three binutils programs (*nm*, *objdump*, and *cxxfilt*) and two image processing programs (*gif2png* and *FFmpeg*) used in prior fuzzing evaluations [9, 44, 45, 55, 58]. We found that experiments that deviate from the above recipe could easily lead one to draw incorrect conclusions, for these reasons:

CCS '18, October 15–19, 2018, Toronto, ON, Canada

ACM ISBN 978-1-4503-5693-0/18/10…$15.00
https://doi.org/10.1145/3243734.3243804

Looking at the data: Attack vectors



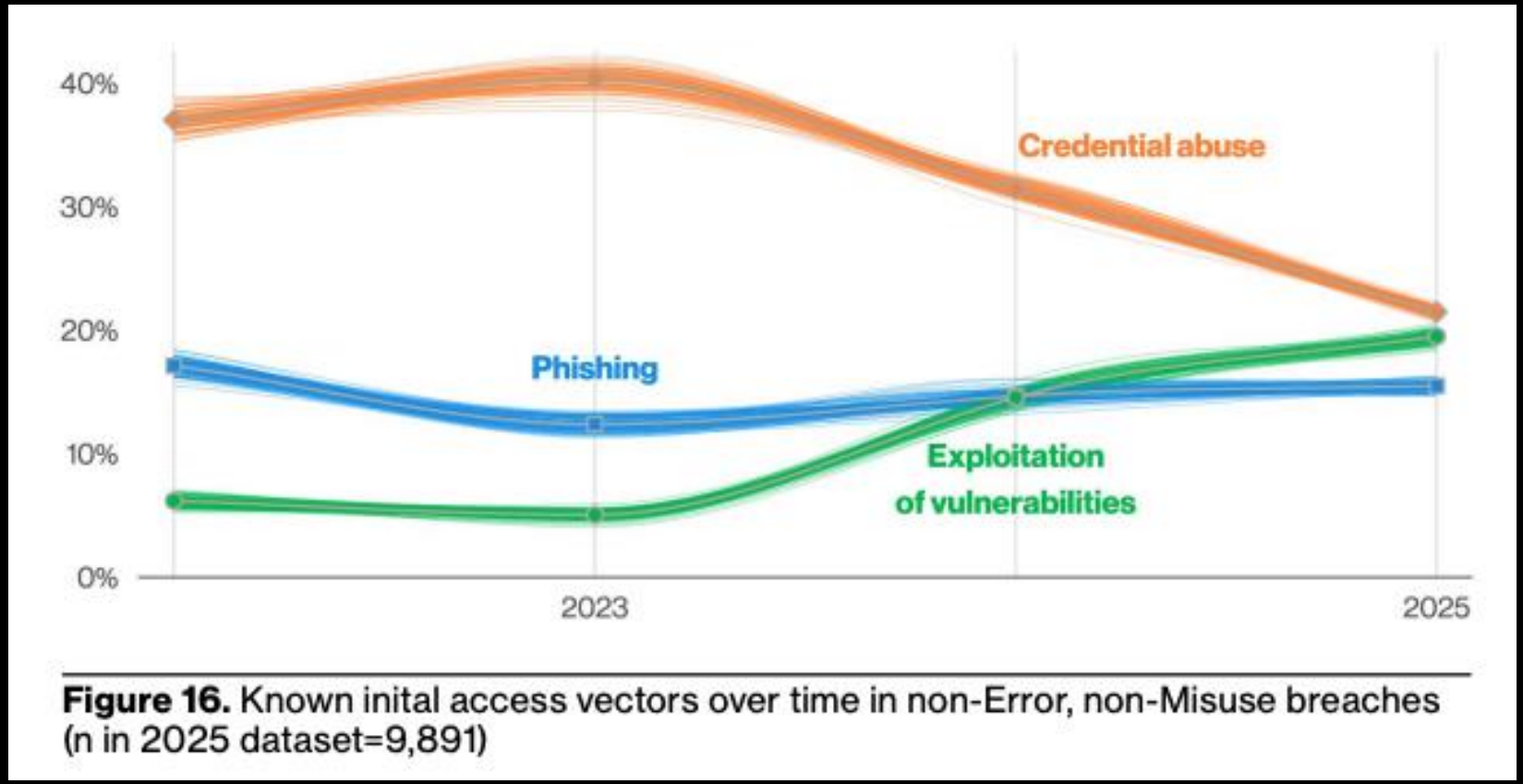
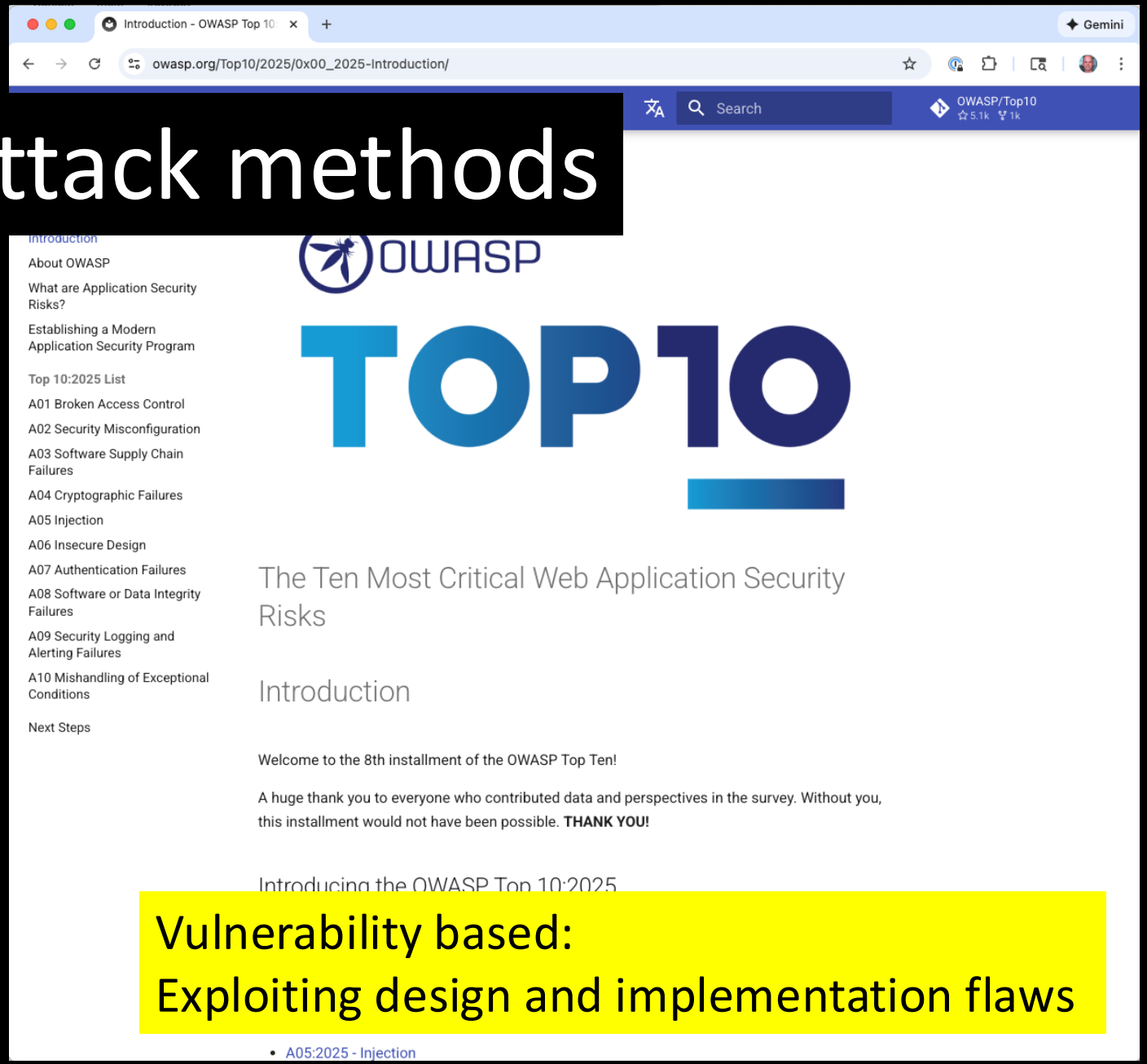


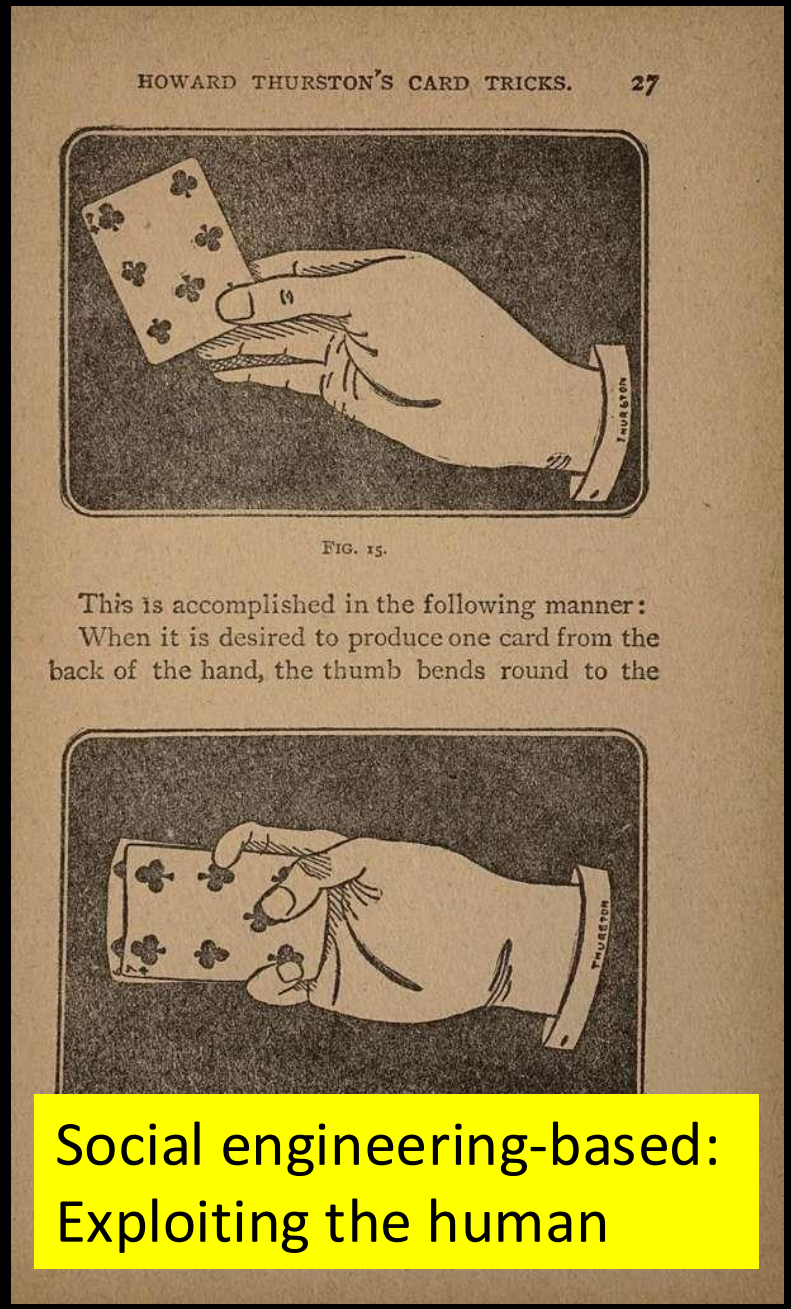
Figure 16. Known inital access vectors over time in non-Error, non-Misuse breaches (n in 2025 dataset=9,891)

Source: 2025 Verizon Data Breach Investigations Report

Attack methods



Vulnerability based:
Exploiting design and implementation flaws



Social engineering-based:
Exploiting the human

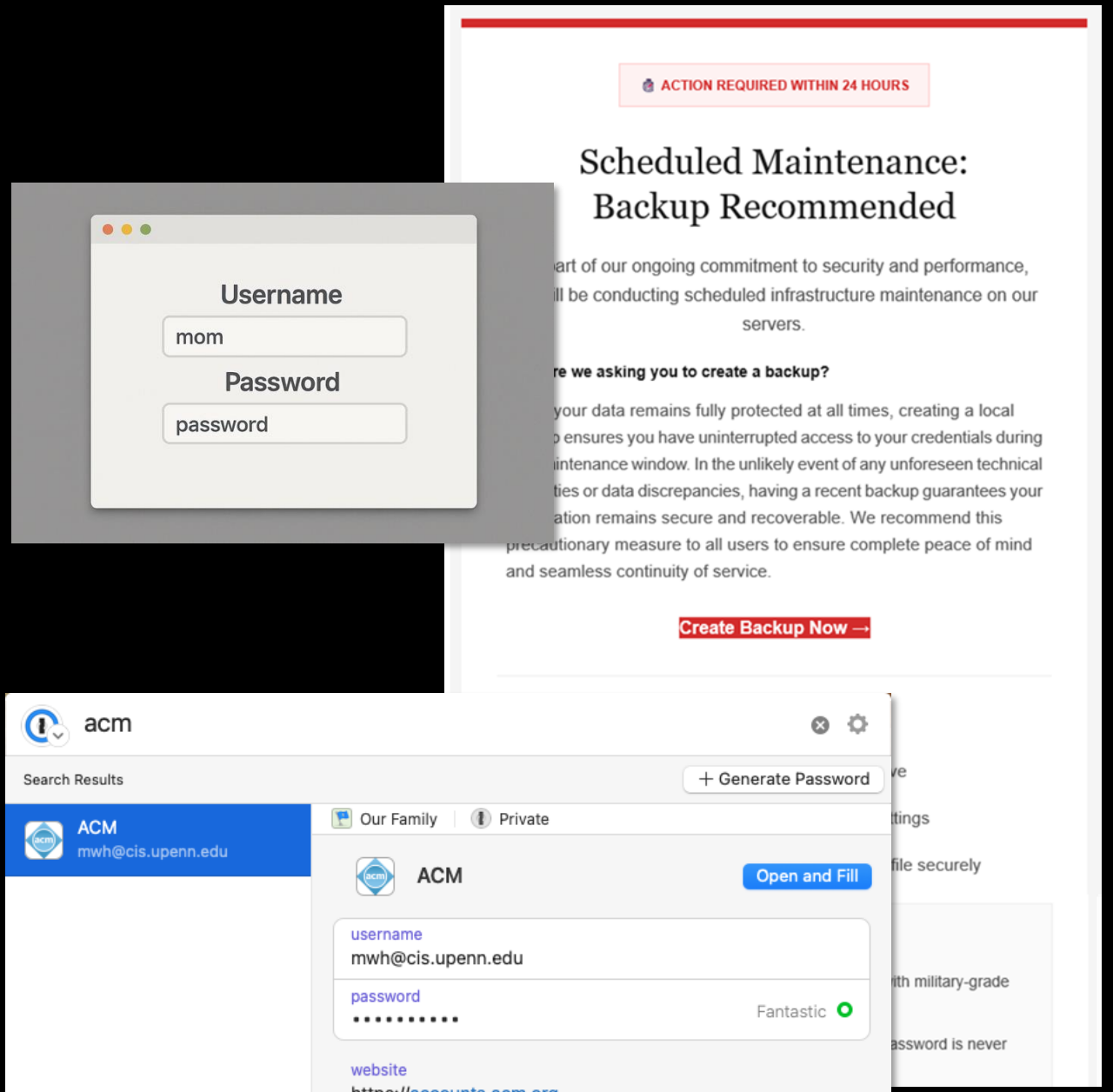
Data source: MITRE Top 25 CWEs

2025 CWE Top 25

Rank	ID	Name	Score	CVEs in KEV	Rank Change vs. 2024
1	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	60.38	7	0
2	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	28.72	4	+1
3	CWE-352	Cross-Site Request Forgery (CSRF)	13.64	0	+1
4	CWE-862	Missing Authorization	13.28	0	+5
5	CWE-787	Out-of-bounds Write	12.68	12	-3
6	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	8.99	10	-1
7	CWE-416	Use After Free	8.47	14	+1
8	CWE-125	Out-of-bounds Read	7.88	3	-2
9	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	7.85	20	-2
10	CWE-94	Improper Control of Generation of Code ('Code Injection')	7.57	7	+1
11	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')	6.96	0	N/A
12	CWE-434	Unrestricted Upload of File with Dangerous Type	6.87	4	-2
13	CWE-476	NULL Pointer Dereference	6.41	0	+8
14	CWE-121	Stack-based Buffer Overflow	5.75	4	N/A
15	CWE-502	Deserialization of Untrusted Data	5.23	11	+1
16	CWE-122	Heap-based Buffer Overflow	5.21	6	N/A
17	CWE-863	Incorrect Authorization	4.14	4	+1
18	CWE-20	Improper Input Validation	4.09	2	-6
19	CWE-284	Improper Access Control	4.07	1	N/A
20	CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	4.01	1	-3
21	CWE-306	Missing Authentication for Critical Function	3.47	11	+4
22	CWE-918	Server-Side Request Forgery (SSRF)	3.36	0	-3
23	CWE-77	Improper Neutralization of Special Elements used in a Command ('Command Injection')	3.15	2	-10
24	CWE-639	Authorization Bypass Through User-Controlled Key	2.62	0	+6
25	CWE-770	Allocation of Resources Without Limits or Throttling	2.54	0	+1

Humans in the loop

- Do they know they are supposed to be doing something?
- Do they understand what they are supposed to do?
- Do they know how to do it?
- Are they motivated to do it?
- Are they capable of doing it?
- Will they actually do it?

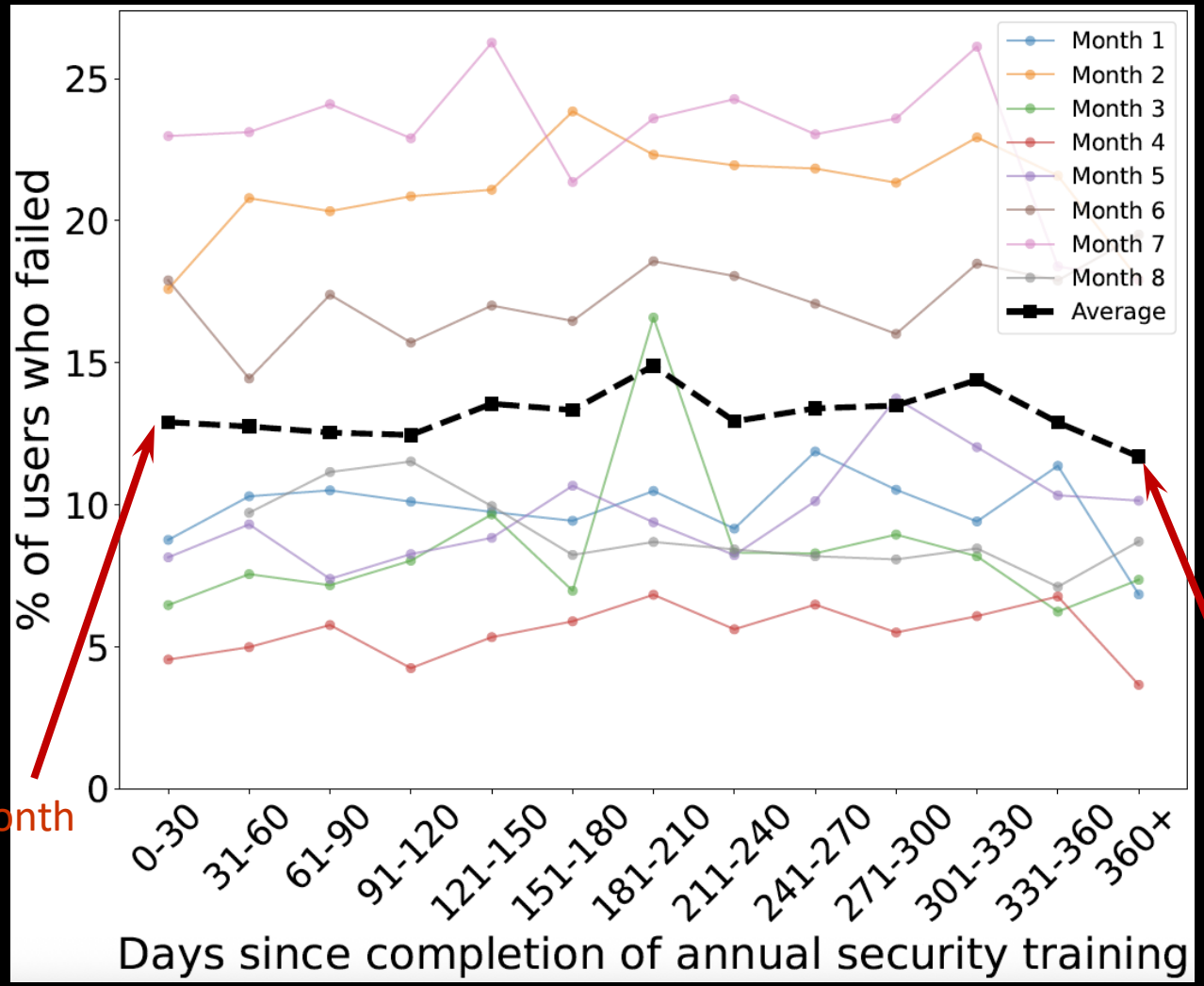


Data: Generic training benefits dubious

Can help address prevalent phishing attacks?

Evidence: Data and statistical models (GLME) find ***no association*** between:

1. how long ago a user completed training (KnowBe4) and
2. their likelihood of failing a phishing simulation



Source: Ho et al, “Understanding the efficacy of phishing training in practice.” S&P 2025.

Remember: Convenience always wins

Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security
- **Follow a well-designed process for secure systems construction**, from threat modeling to building to testing to maintenance

All while taking a data-informed approach

Secure by Design | CISA

Gemini

cisa.gov/securebydesign

Secure by Design

It's time to build cybersecurity into the design and manufacture of technology products.

As America's cyber defense agency, CISA is charged with defending our nation against ever-evolving cyber threats and to understand, manage, and reduce risk to the cyber and physical infrastructure that Americans rely on every hour of every day. But, as we introduce more unsafe techno

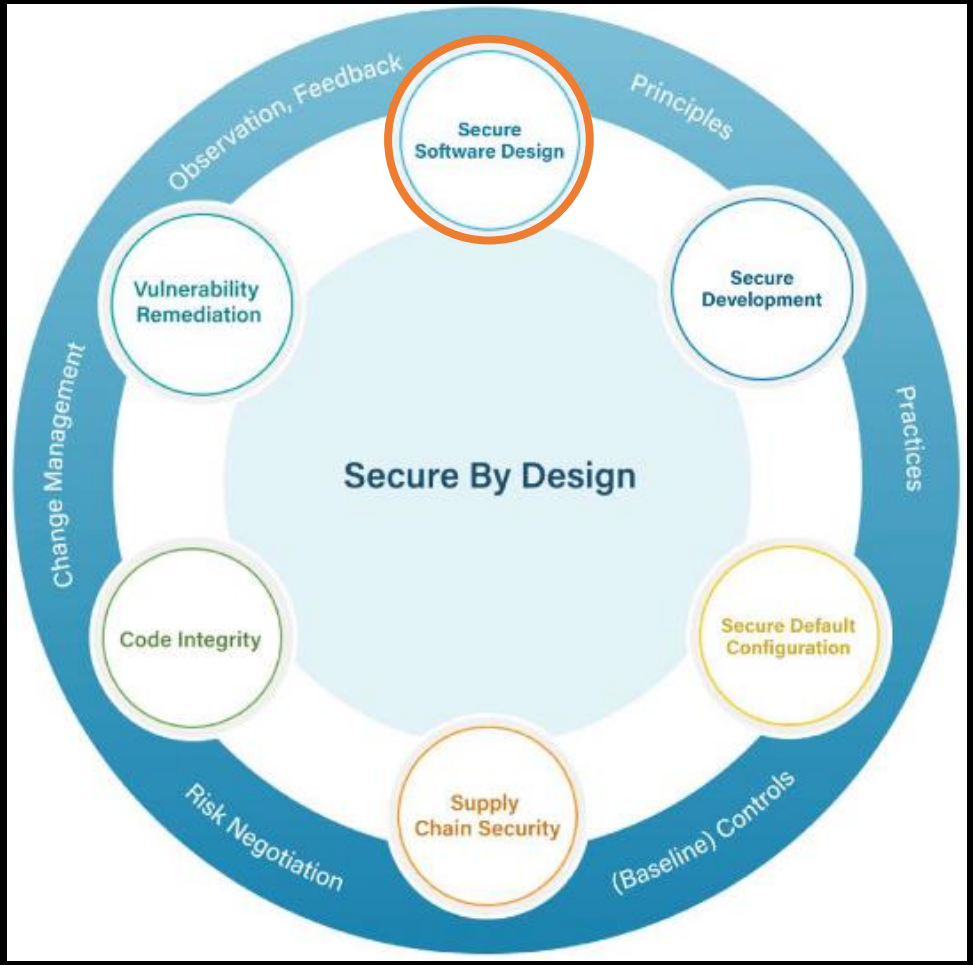
As a nation, we have allowed a system where the cy consumers and small organizations and away from that increasingly run our digital lives. Americans ne consumers can trust the safety and integrity of the t

> Every technology provider must take ownership at the executive level to ensure their products are secure by design.

"prioritize the security of customers as a core business requirement" and "implement Secure by Design principles to significantly decrease the number of exploitable flaws."

Six Considerations: Overview

1. Secure Software Design
2. Secure Development
3. Secure Default Configuration

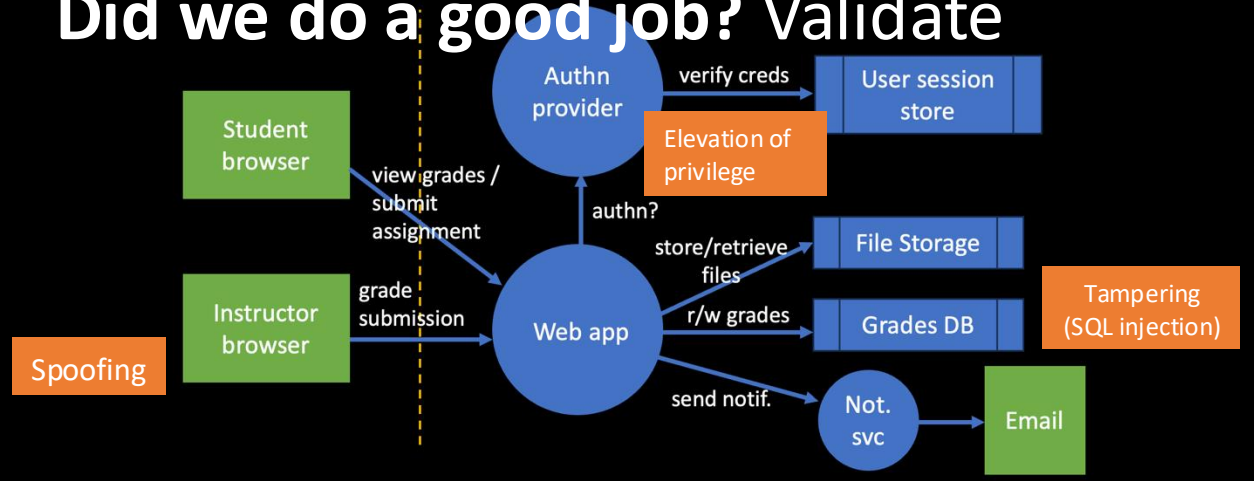


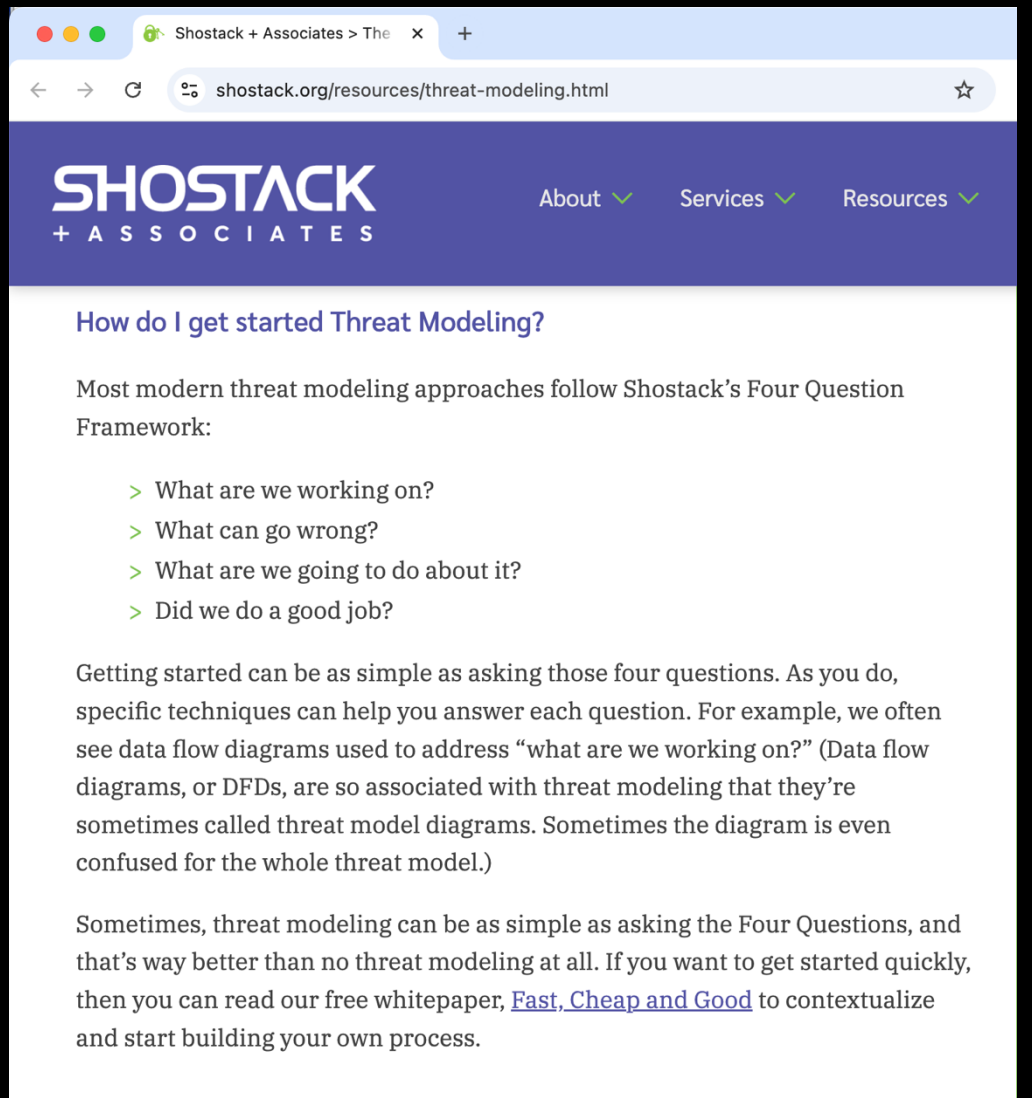
4. Supply Chain Security
5. Code Integrity
6. Vulnerability Remediation

The Four Questions of Threat Modeling

Shostack's four-question framework

1. **What are we working on?** Describe the system
2. **What can go wrong?** Identify threats
3. **What are we going to do about it?** Mitigate, accept, transfer, eliminate
4. **Did we do a good job?** Validate





Secure Design: Properties, Principles, Controls

- **Principles** and **Controls**
- **Confidentiality** (by encryption)
- **Authentication** – proving identity
- **Authorization**
- **Integrity** and **Accountability**

- Principles:
  - **Do not expect expert users**
  - **Fail-safe defaults**
  - **Favor simplicity**
  - **Defend in depth** (throughout)
  - **Trust with Reluctance** (all)
  - **Monitoring and Traceability**
  - **Putting it all together**
  - **Use community resources**

The Ecosystem Thesis

"The safety and security of a software application or service is substantially an **emergent property of the developer ecosystem** that produced it."

To improve security, redesign the *ecosystem*, not just the guidance.

Programming languages, libraries, frameworks, build tooling, deployment infrastructure, configuration surfaces.

practice

DOI:10.1145/3651621

Article development led by acmqueue
queue.acm.org

Continuous assurance at scale.

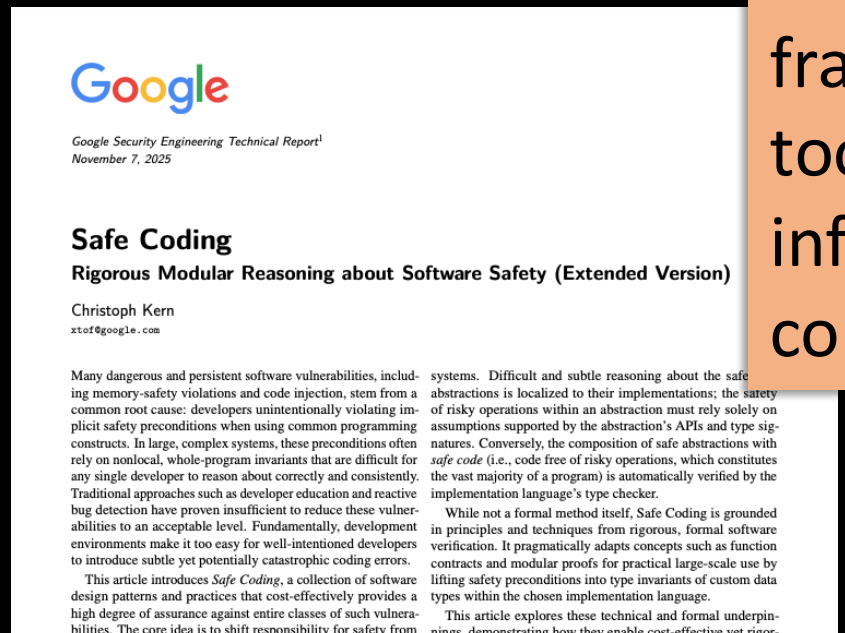
BY CHRISTOPH KERN

Developer Ecosystems for Software Safety

Security Project (OWASP) Cheat Sheet Series.[b]

Despite these efforts, common types of software defects prevail, and many occupy top ranks of "worst vulnerabilities" lists such as the OWASP Top 10[c] or the CWE Top 25 Most Dangerous Software Weaknesses[d] for years if not decades.

Based on work at Google over the past decade on managing the risk of software defects in its wide-ranging portfolio of applications and services, the members of Google's security engineering team developed a theory about the reason for the prevalence of defects: It's simply too difficult for real-world development and operations teams to apply the available guidance comprehensively and consistently, which results in a problematic rate of new defects. Commonly used approaches to find and fix implementation defects after the fact can help (for example, code review, testing, scanning, or static and dynamic analysis such as fuzzing), but in practice they find only a fraction of these defects. Design-level defects are difficult or impractical to remediate



Google

Google Security Engineering Technical Report[1]
November 7, 2025

Safe Coding

Rigorous Modular Reasoning about Software Safety (Extended Version)

Christoph Kern
xtof@google.com

Many dangerous and persistent software vulnerabilities, including memory-safety violations and code injection, stem from a common root cause: developers unintentionally violating implicit safety preconditions when using common programming constructs. In large, complex systems, these preconditions often rely on nonlocal, whole-program invariants that are difficult for any single developer to reason about correctly and consistently. Traditional approaches such as developer education and reactive bug detection have proven insufficient to reduce these vulnerabilities to an acceptable level. Fundamentally, development environments make it too easy for well-intentioned developers to introduce subtle yet potentially catastrophic coding errors.

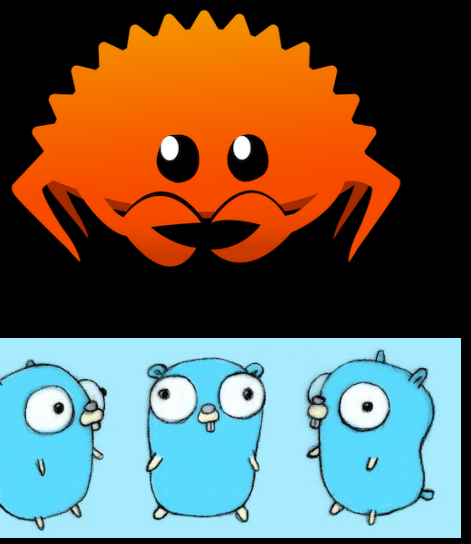
This article introduces *Safe Coding*, a collection of software design patterns and practices that cost-effectively provides a high degree of assurance against entire classes of such vulnera-

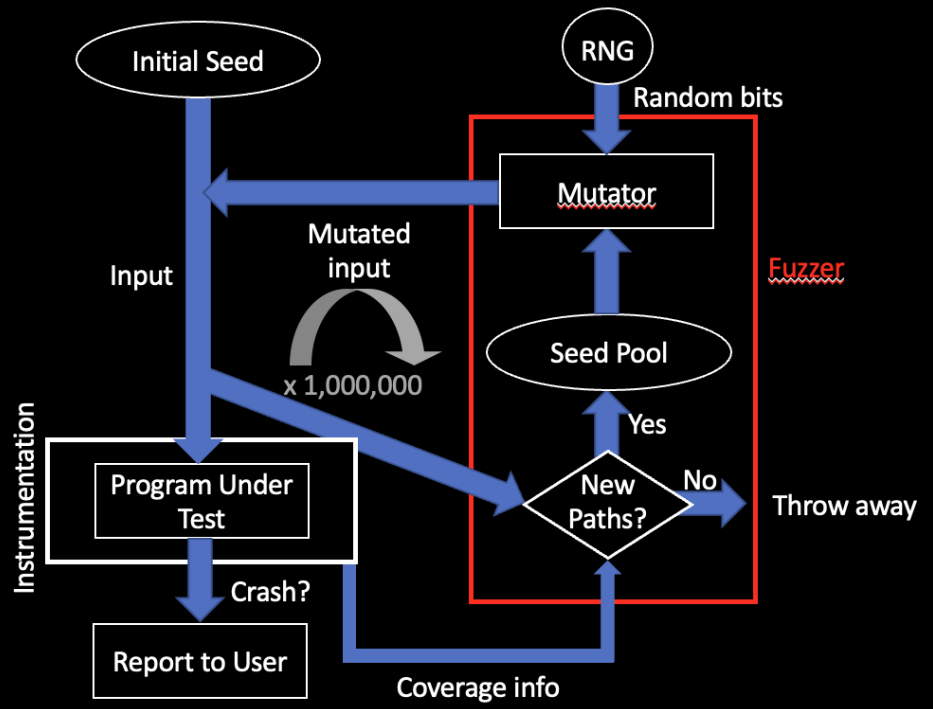
systems. Difficult and subtle reasoning about the safe abstractions is localized to their implementations; the safety of risky operations within an abstraction must rely solely on assumptions supported by the abstraction's APIs and type signatures. Conversely, the composition of safe abstractions with *safe code* (i.e., code free of risky operations, which constitutes the vast majority of a program) is automatically verified by the implementation language's type checker.

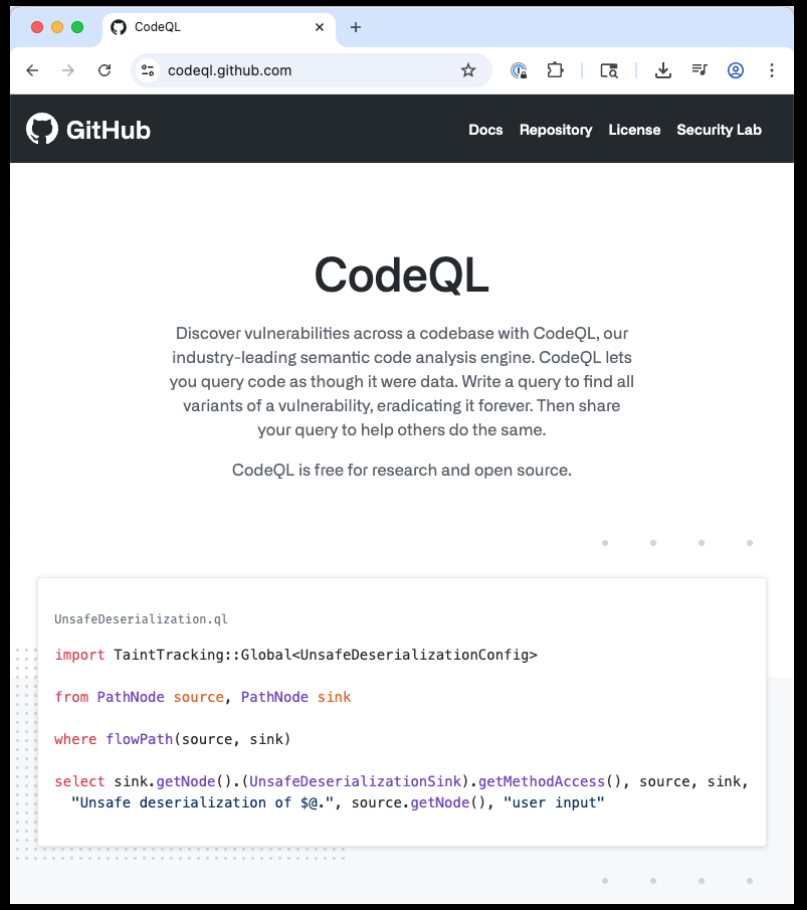
While not a formal method itself, Safe Coding is grounded in principles and techniques from rigorous, formal software verification. It pragmatically adapts concepts such as function contracts and modular proofs for practical large-scale use by lifting safety preconditions into type invariants of custom data types within the chosen implementation language.

This article explores these technical and formal underpin-

Secure Dev: Safe PLs, fuzzers, analyzers, …

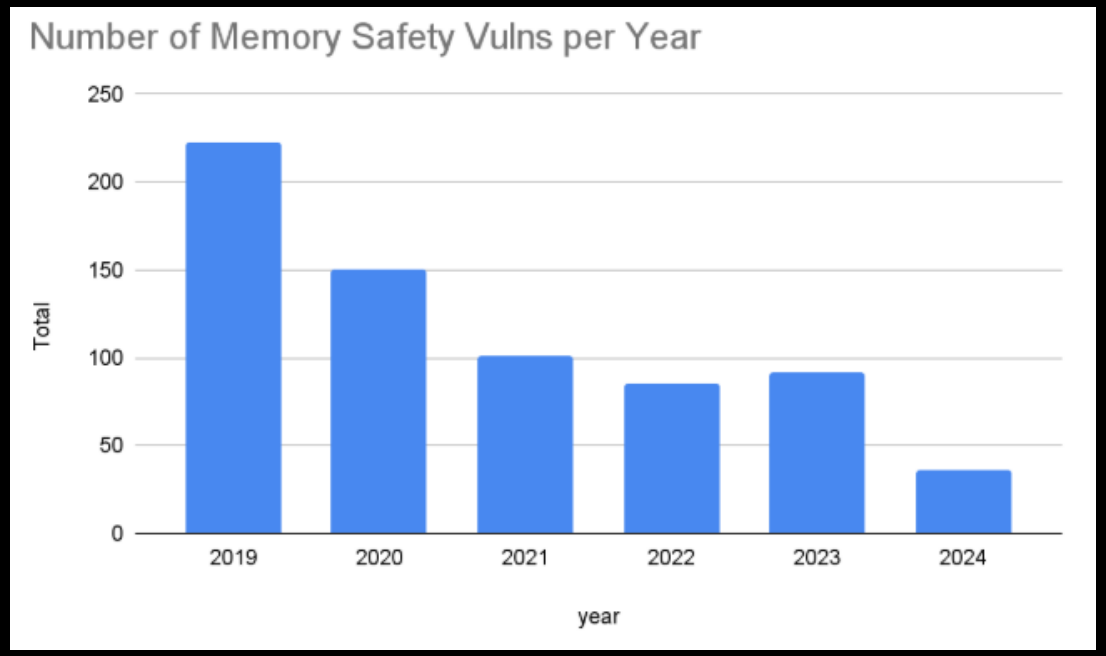


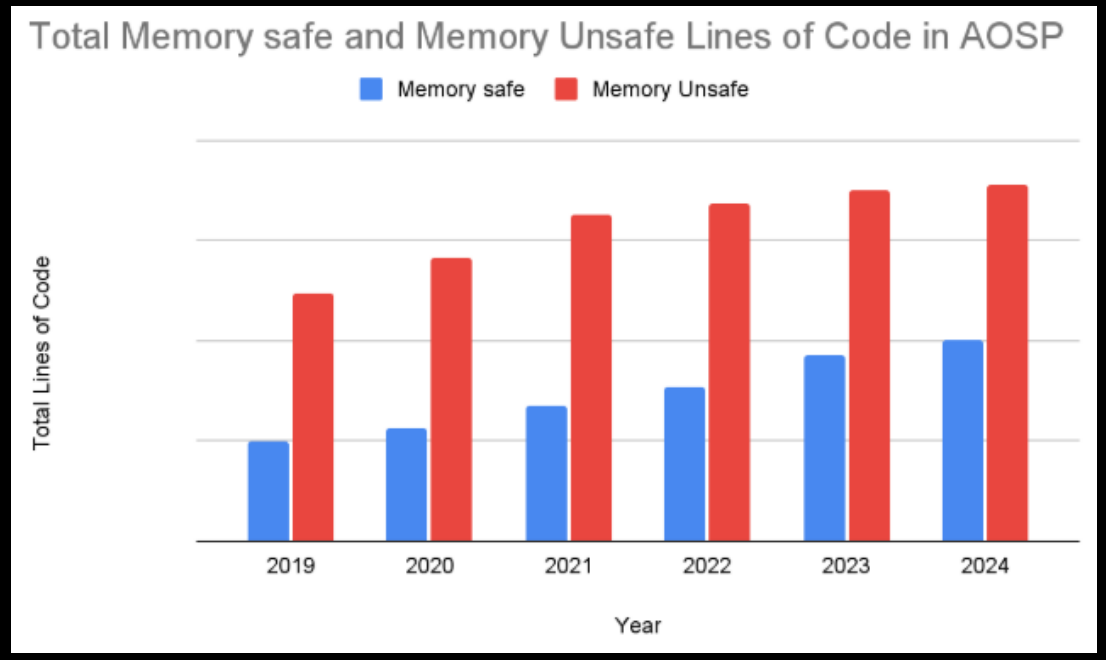




Data: Use of PL can prevent vulnerabilities

Android is writing most new code in Rust, and fixing vulns in its C/C++.
Result: A roughly exponential drop in vulnerabilities reported

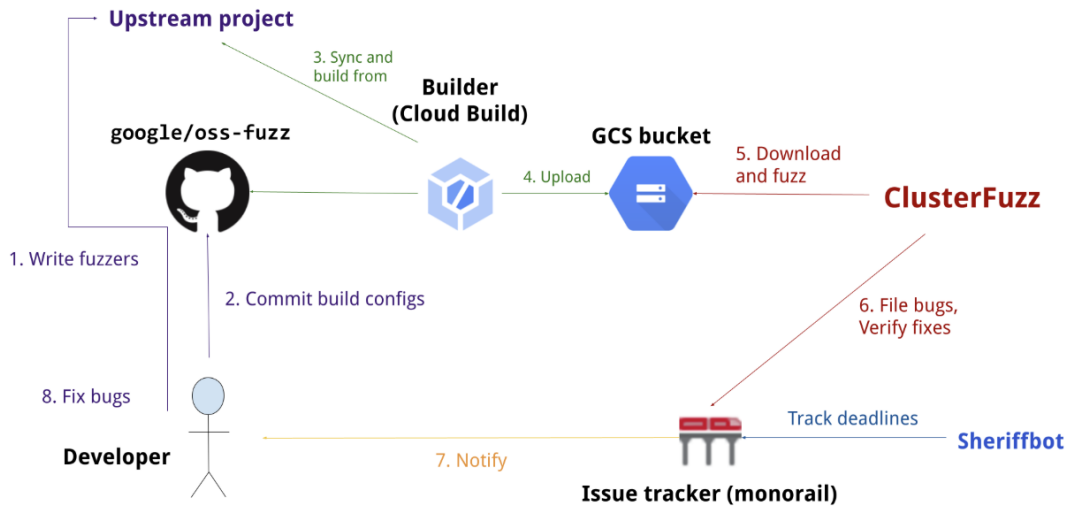




Source: Google Security Blog

OSS-Fuzz

Overview



Documentation

Read our detailed documentation to learn how to use OSS-Fuzz.

Trophies

As of May 2025, OSS-Fuzz has helped identify and fix over 13,000 vulnerabilities and 50,000 bugs across 1,000 projects.

OSS-Fuzz | Documentation fo

Gemini

google.github.io/oss-fuzz/

OSS-Fuzz

Search OSS-Fuzz

OSS-Fuzz on GitHub

OSS-Fuzz

ture

arted

topics

ading

guidance

es Just the Docs, a

tion theme for Jekyll.

https://en.wikipedia.org/wiki/Buffer_overflow

OSS-Fuzz

Fuzz testing is a well-known technique for uncovering programming errors in software. Many of these detectable errors, like buffer overflow, can have serious security implications. Google has found thousands of security vulnerabilities and stability bugs by deploying guided in-process fuzzing of Chrome components, and we now want to share that service with the open source community.

In cooperation with the Core Infrastructure Initiative and the OpenSSF, OSS-Fuzz aims to make common open source software more secure and stable by combining modern fuzzing techniques with scalable, distributed execution. Projects that do not qualify for OSS-Fuzz (e.g. closed source) can run their own instances of ClusterFuzz or ClusterFuzzLite.

We support the libFuzzer, AFL++, Honggfuzz, and Centipede fuzzing engines in combination with Sanitizers, as well as ClusterFuzz, a distributed fuzzer execution environment and reporting tool.

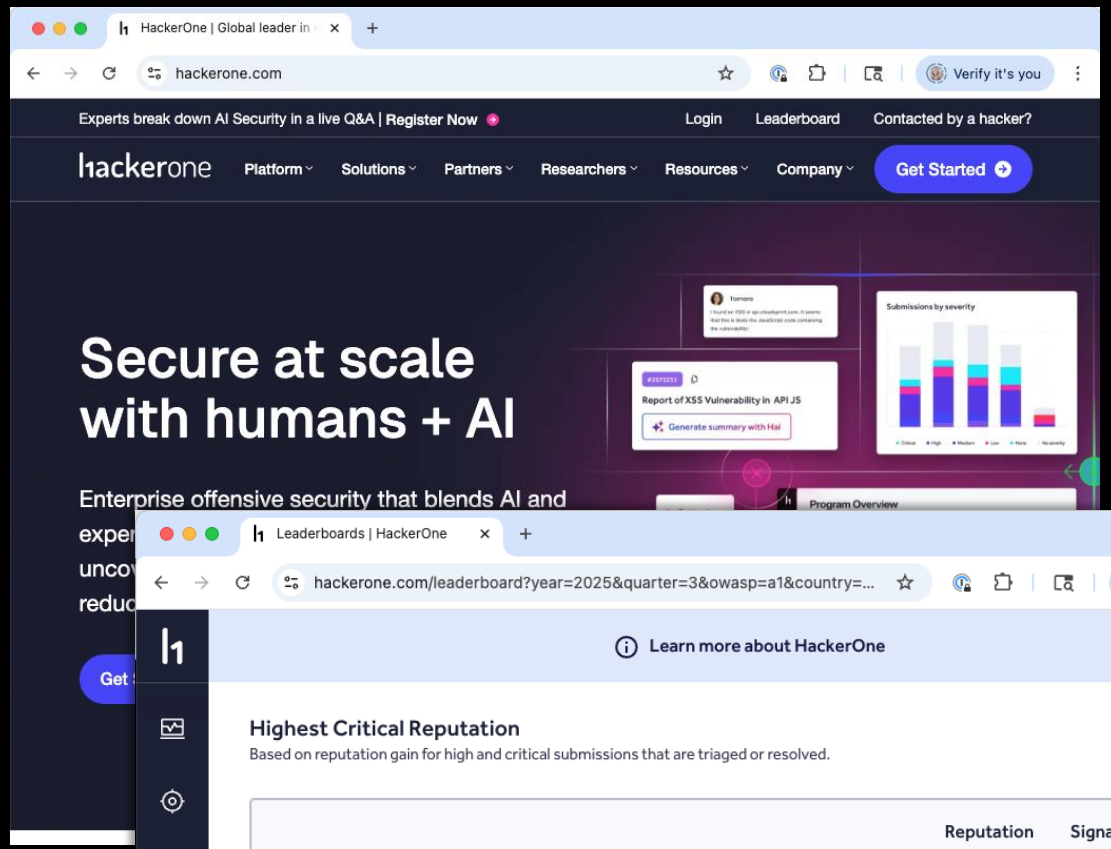
Currently, OSS-Fuzz supports C/C++, Rust, Go, Python and Java/JVM code. Other languages supported by LLVM may work too. OSS-Fuzz supports fuzzing x86_64 and i386 builds.

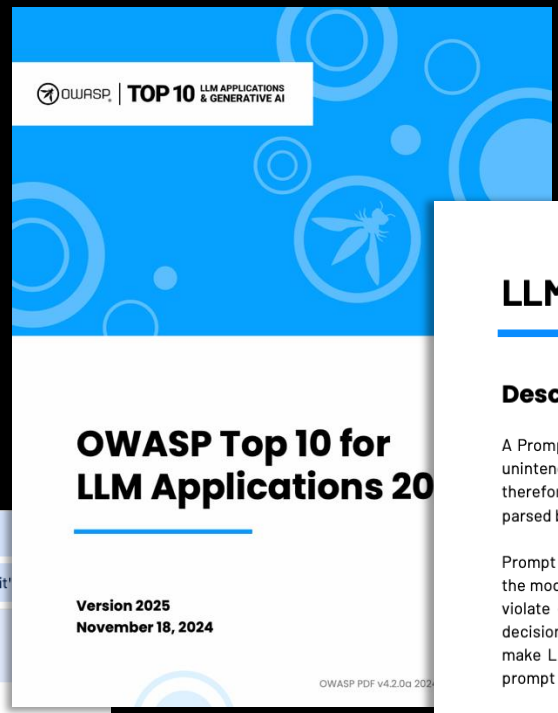
Project history

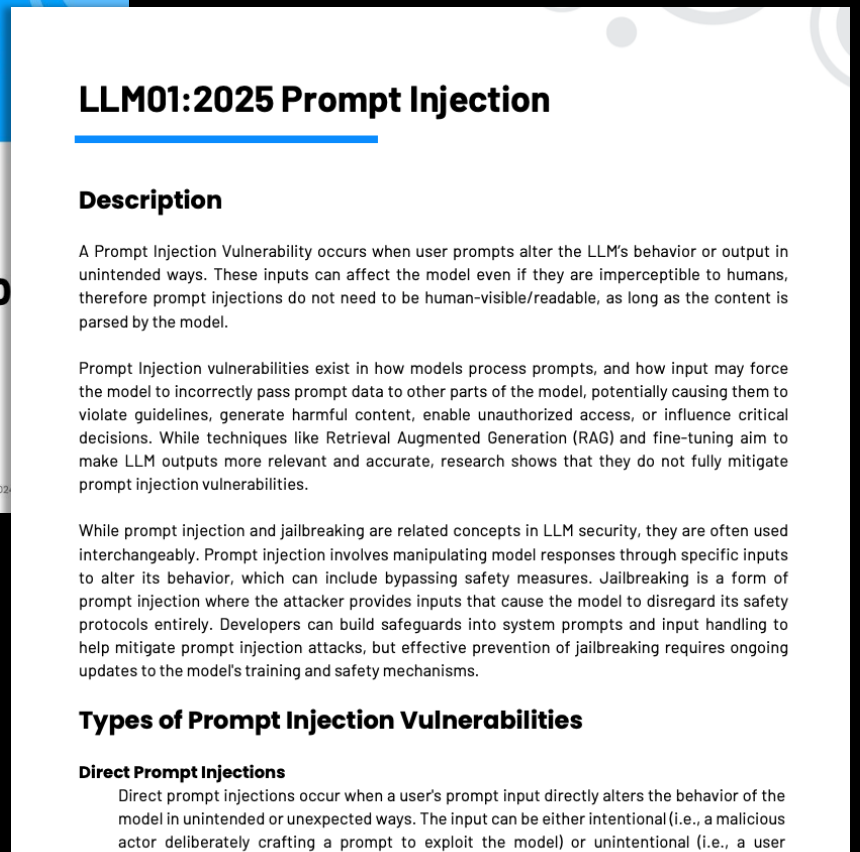
OSS-Fuzz was launched in 2016 in response to the Heartbleed vulnerability, discovered in OpenSSL, one of the most popular open source projects for encrypting web traffic. The vulnerability had the potential to affect almost every internet user, yet was caused by a relatively simple memory buffer overflow bug that could have been detected by fuzzing—that is, by running the code on randomized inputs to intentionally cause unexpected behaviors or crashes. At the time, though, fuzzing was not widely used and was cumbersome for developers, requiring extensive manual effort.

Google created OSS-Fuzz to fill this gap: it's a free service that runs fuzzers for open source

LLMs and GenAI: Game Changers







LLM01:2025 Prompt Injection

Description

A Prompt Injection Vulnerability occurs when user prompts alter the LLM's behavior or output in unintended ways. These inputs can affect the model even if they are imperceptible to humans, therefore prompt injections do not need to be human-visible/readable, as long as the content is parsed by the model.

Prompt Injection vulnerabilities exist in how models process prompts, and how input may force the model to incorrectly pass prompt data to other parts of the model, potentially causing them to violate guidelines, generate harmful content, enable unauthorized access, or influence critical decisions. While techniques like Retrieval Augmented Generation (RAG) and fine-tuning aim to make LLM outputs more relevant and accurate, research shows that they do not fully mitigate prompt injection vulnerabilities.

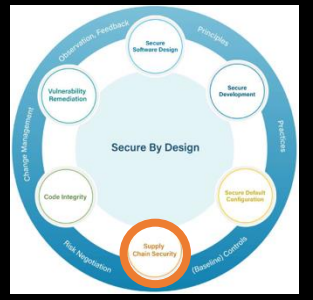
While prompt injection and jailbreaking are related concepts in LLM security, they are often used interchangeably. Prompt injection involves manipulating model responses through specific inputs to alter its behavior, which can include bypassing safety measures. Jailbreaking is a form of prompt injection where the attacker provides inputs that cause the model to disregard its safety protocols entirely. Developers can build safeguards into system prompts and input handling to help mitigate prompt injection attacks, but effective prevention of jailbreaking requires ongoing updates to the model's training and safety mechanisms.

Types of Prompt Injection Vulnerabilities

Direct Prompt Injections

Direct prompt injections occur when a user's prompt input directly alters the behavior of the model in unintended or unexpected ways. The input can be either intentional (i.e., a malicious actor deliberately crafting a prompt to exploit the model) or unintentional (i.e., a user

Supply Chain: What's in your software?

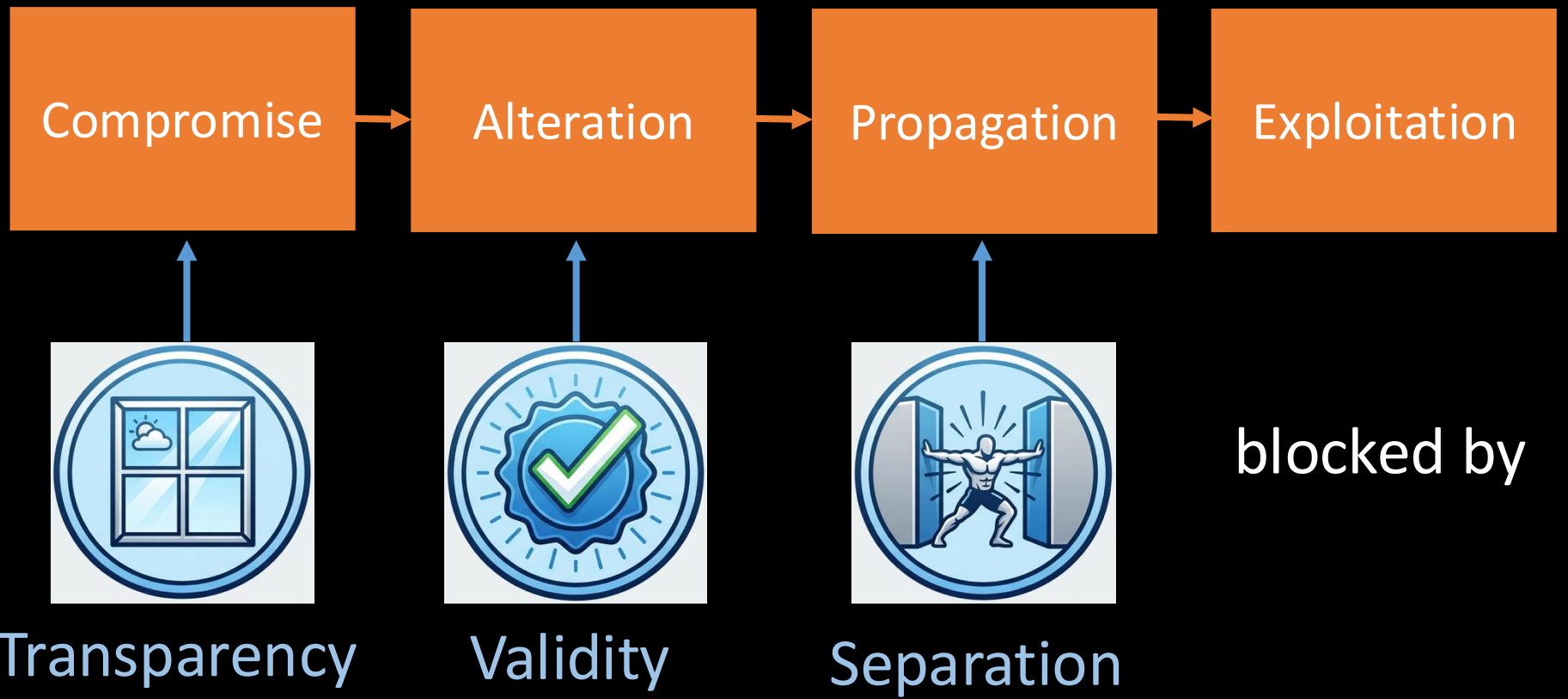


Modern software is **mostly code you didn't write**.

Each dependency is a **trust decision**:

- Is it actively maintained?
- Has it been audited for security?
- What happens when a vulnerability is found in it?

Supply Chain *Attacks*



What is Code Integrity?

Ensuring that the software delivered to users is the software the organization intended to deliver

Example supply chain attack vectors:

1. **Misplaced/stolen trust** —
   malicious actor commits malware or vulnerable code
   - XZ Utils attack as a case study
2. **Tampering** — malicious actor compromises build system, artifacts

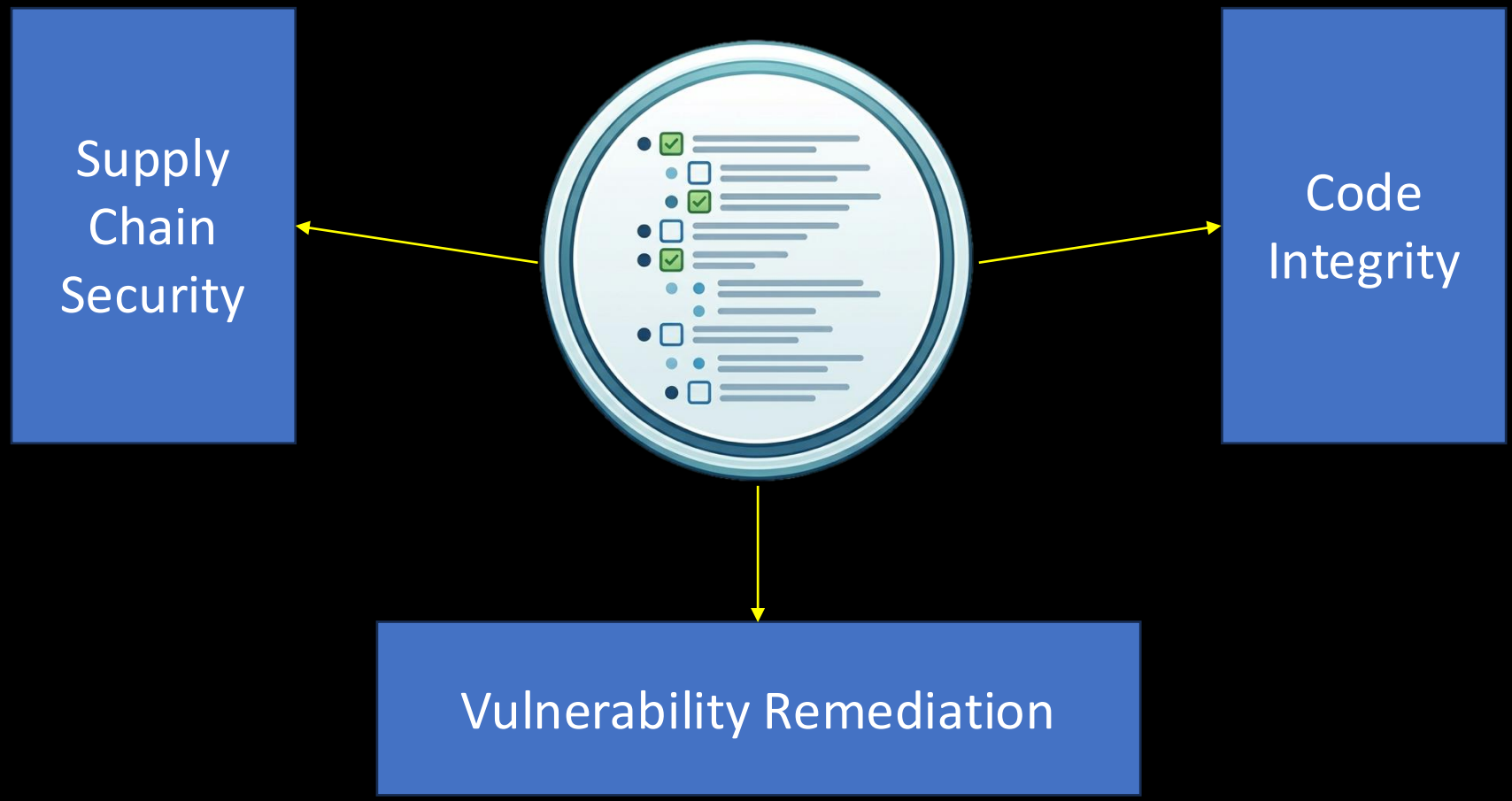
I Found a Vulnerability: Now What?

No software ships bug-free. The question is: **what happens when bugs are found?**

Three phases:

1. **Immediate remediation:** Fix the reported vulnerability, ship a patch
2. **Variant analysis:** Search for *similar* vulnerabilities proactively
3. **Process improvement:** Update tools, training, and practices to prevent recurrence

SBOMs Enable the Three Activities

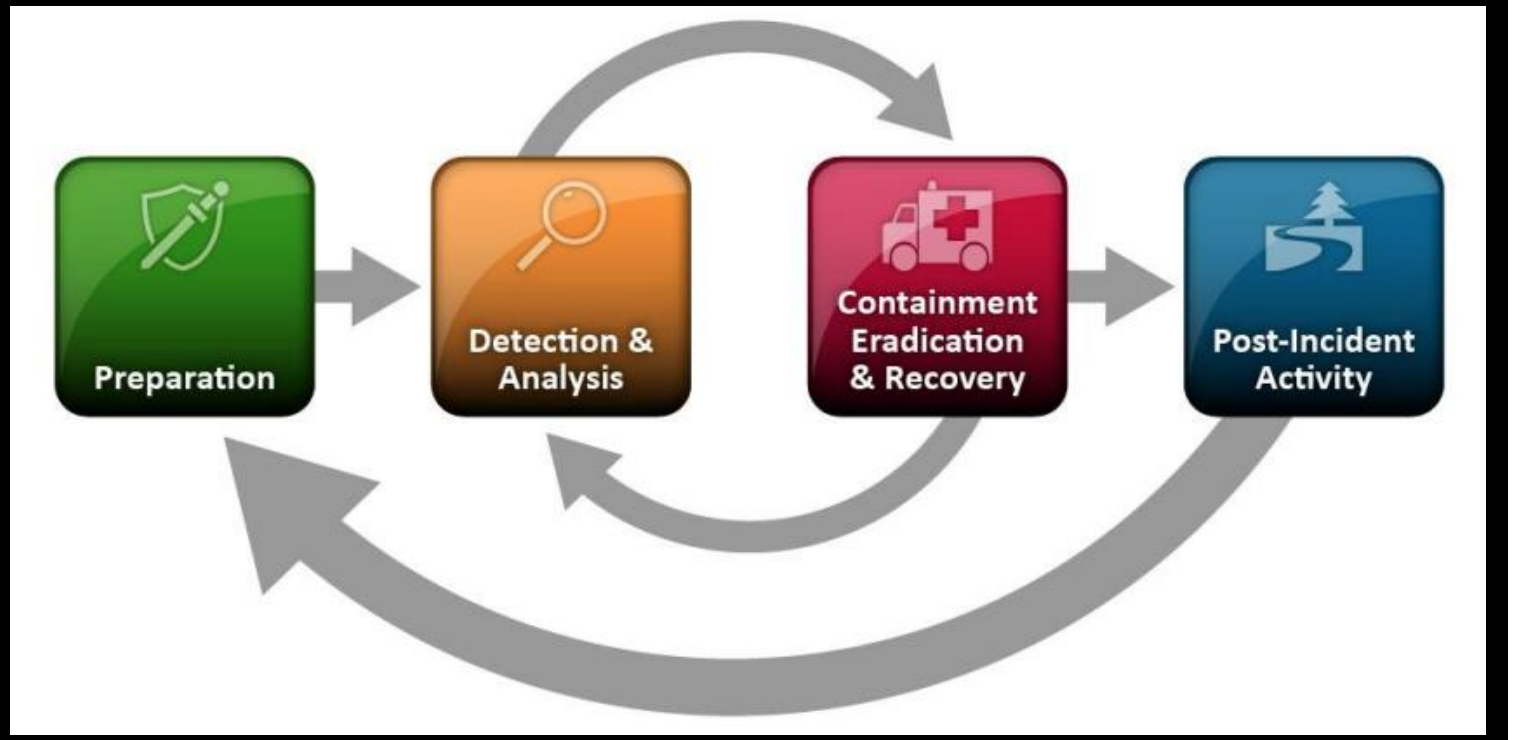


Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security
- **Follow a well-designed process for secure systems construction**, from threat modeling to building to testing to maintenance
- **Manage security operations** – preventing, detecting, mitigating, and recovering from incidents – and gather data to improve future posture

All while taking a data-informed approach

The Incident Lifecycle (NIST SP 800-61)



NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Special Publication 800-61
Revision 2

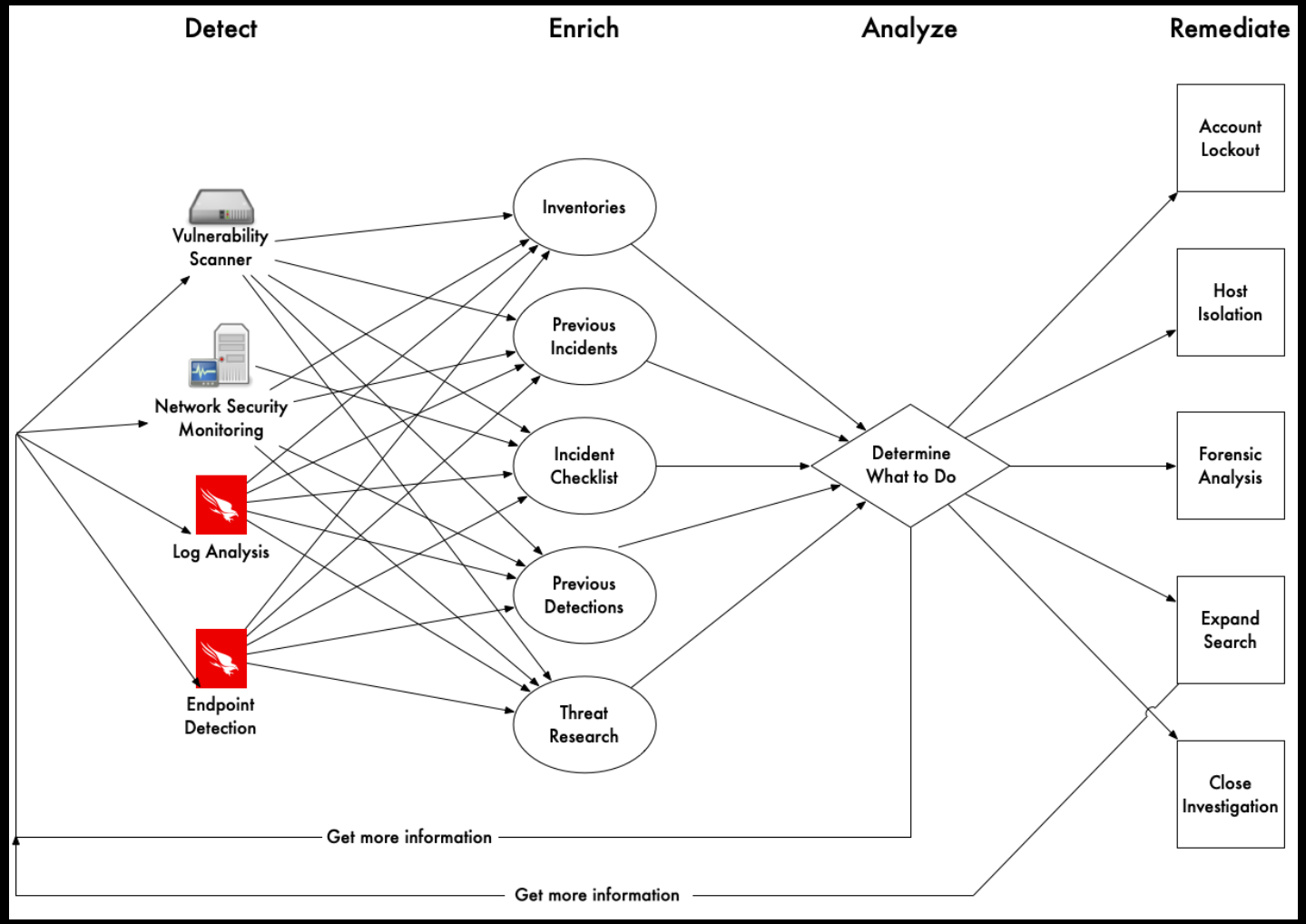
Computer Security Incident Handling Guide

Recommendations of the National Institute of Standards and Technology

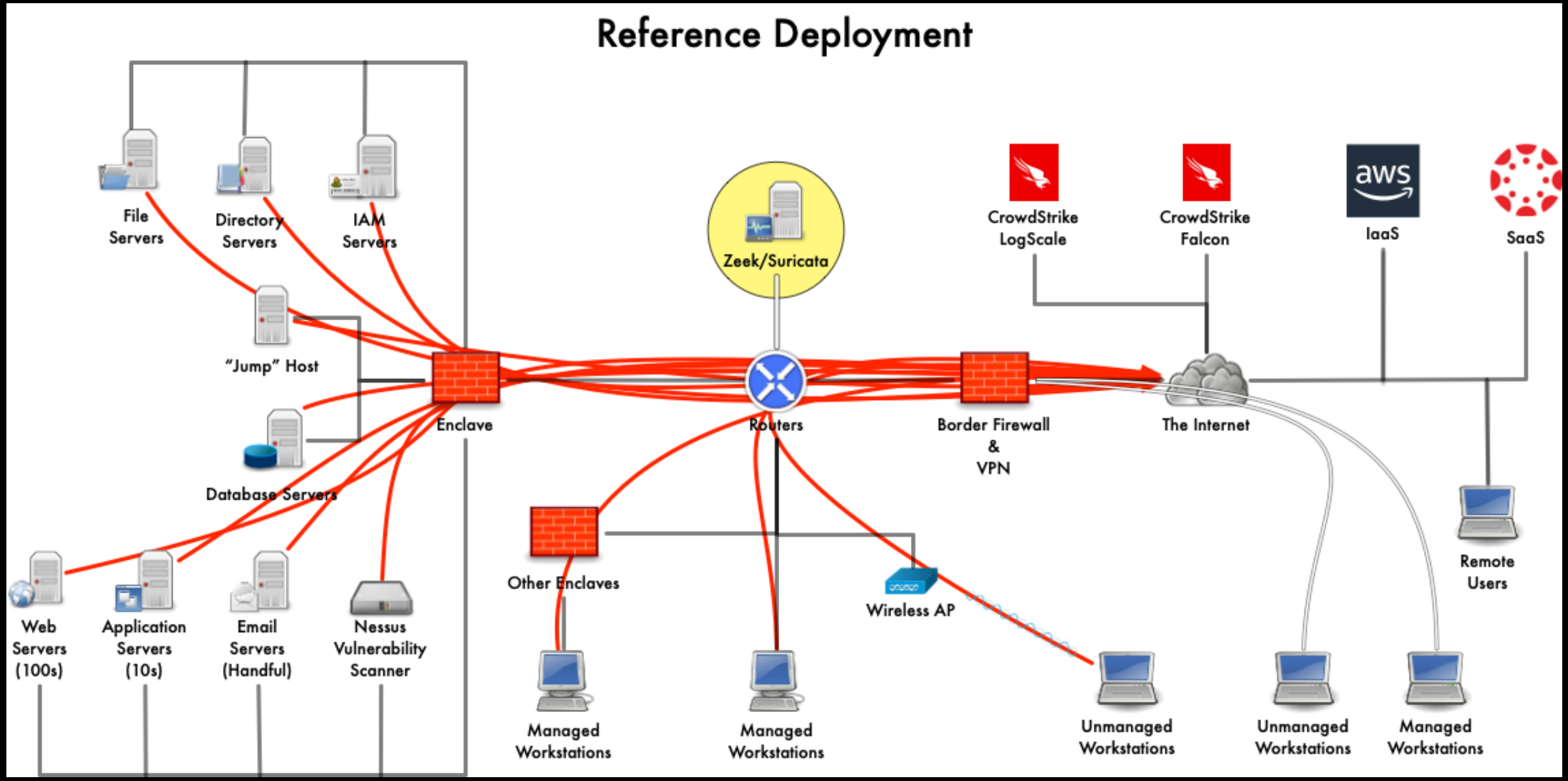
Paul Cichonski
Tom Millar
Tim Grance
Karen Scarfone

http://dx.doi.org/10.6028/NIST.SP.800-61r2

Tools & Inventories = Workflows



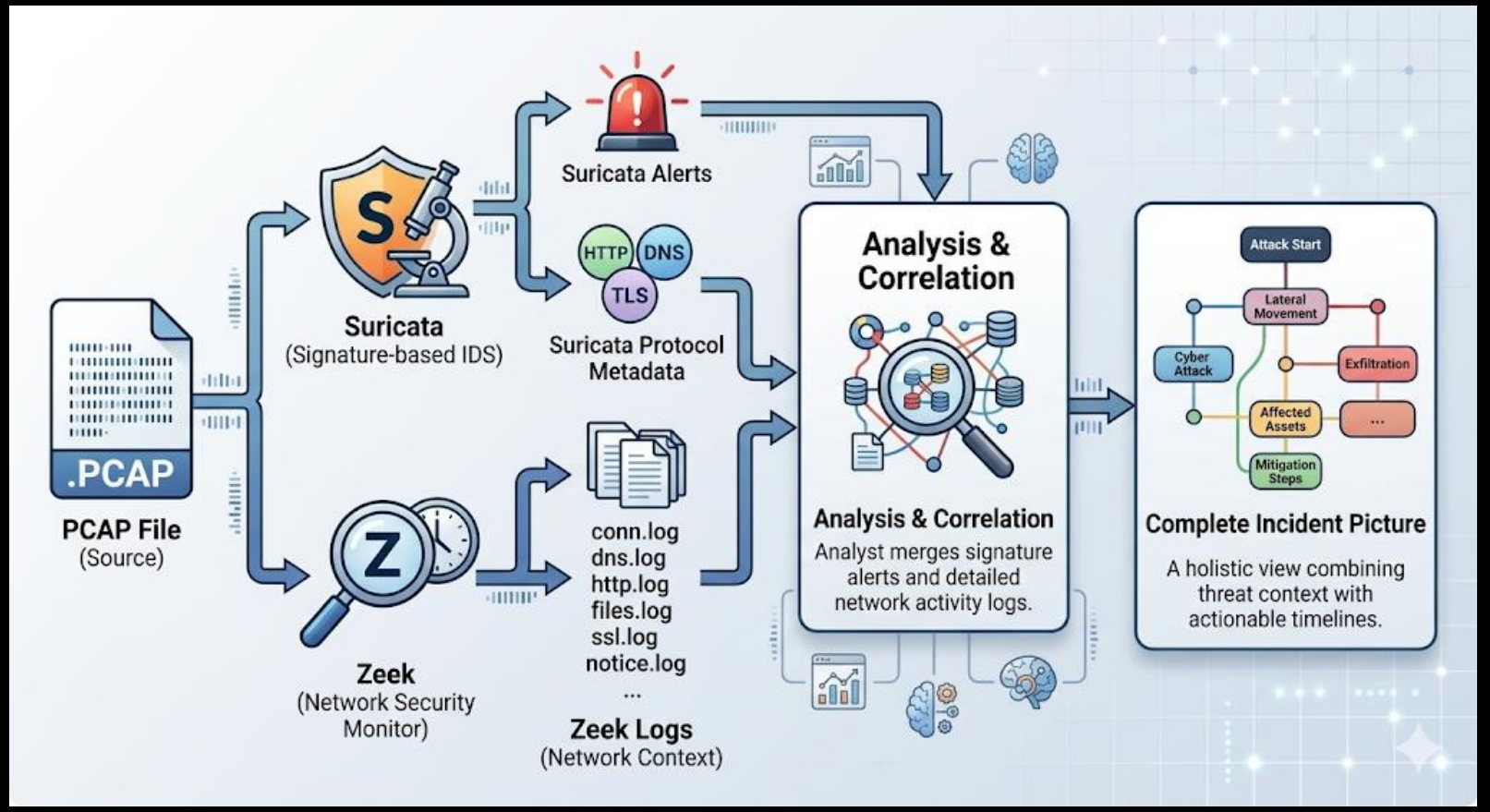
Reference Deployment



Suricata + Zeek: Better Together

Suricata tells you **“something bad happened”**

Zeek tells you **“here’s everything that happened before, during, and after”**



`zeek-cut` useful for extracting specific fields from Zeek's tab-separated logs

Security Operations Centers

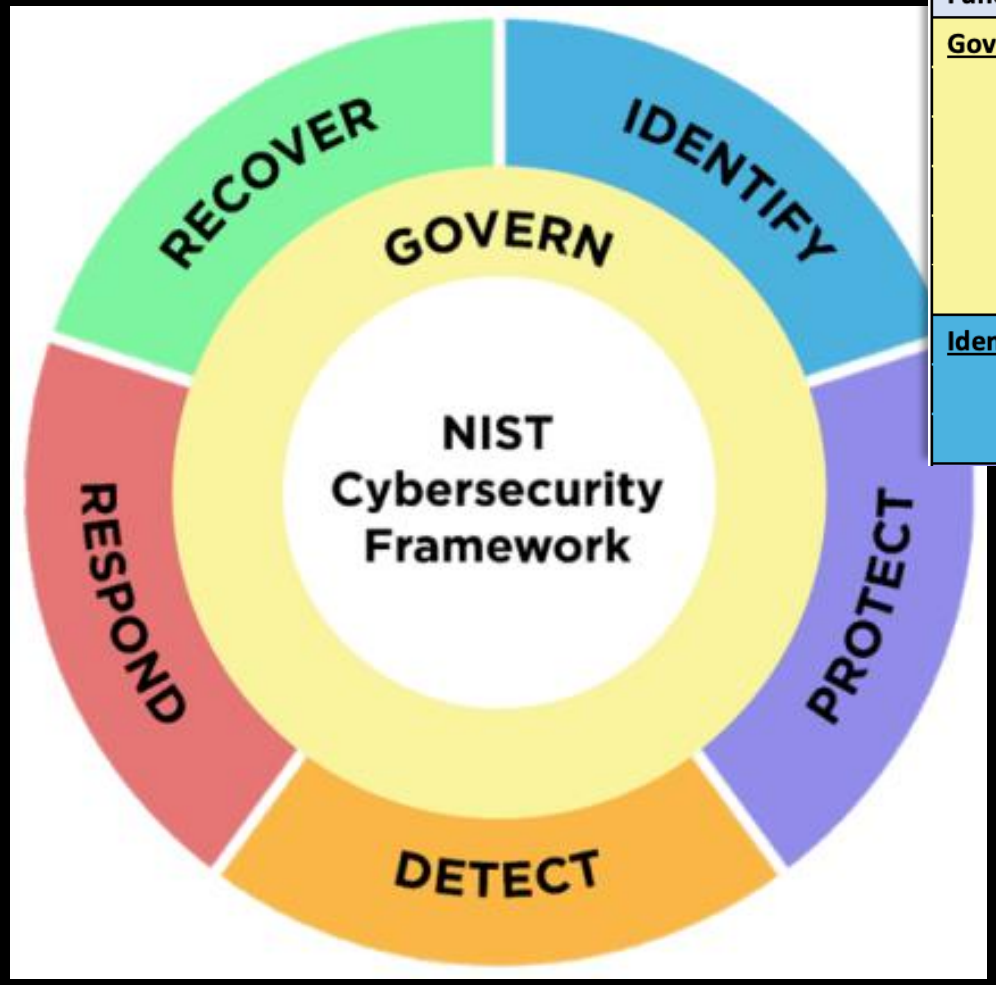
Beware: Alert Fatigue!

Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security
- **Follow a well-designed process for secure systems construction**, from threat modeling to building to testing to maintenance
- **Manage security operations** – preventing, detecting, mitigating, and recovering from incidents – and gather data to improve future posture
- **Make risk-informed decisions**: Assess designs and technologies according to how they mitigate security risk, while leveraging **insurance** and responding to **regulation**

All while taking a data-informed approach

NIST CSF 2.0: What You *Should* Do



Function	Category	Category Identifier
Govern (GV)	Organizational Context	GV.OC
	Risk Management Strategy	GV.RM
	Roles, Responsibilities, and Authorities	GV.RR
	Policy	GV.PO
	Oversight	GV.OV
	Cybersecurity Supply Chain Risk Management	GV.SC
Identify (ID)	Asset Management	ID.AM
	Risk Assessment	ID.RA
	Improvement	ID.IM

Under **Govern** → **Risk Management (GV.RM)**:

- GV.RM-02: Establish **risk appetite and tolerance**
- GV.RM-06: Establish a standardized method for **calculating and prioritizing** risks

Under **Identify** → **Risk Assessment (ID.RA)**:

- ID.RA-03: Identify and record threats
- ID.RA-04: Identify potential **impacts and likelihoods**
- ID.RA-05: Use these to **prioritize** risk responses

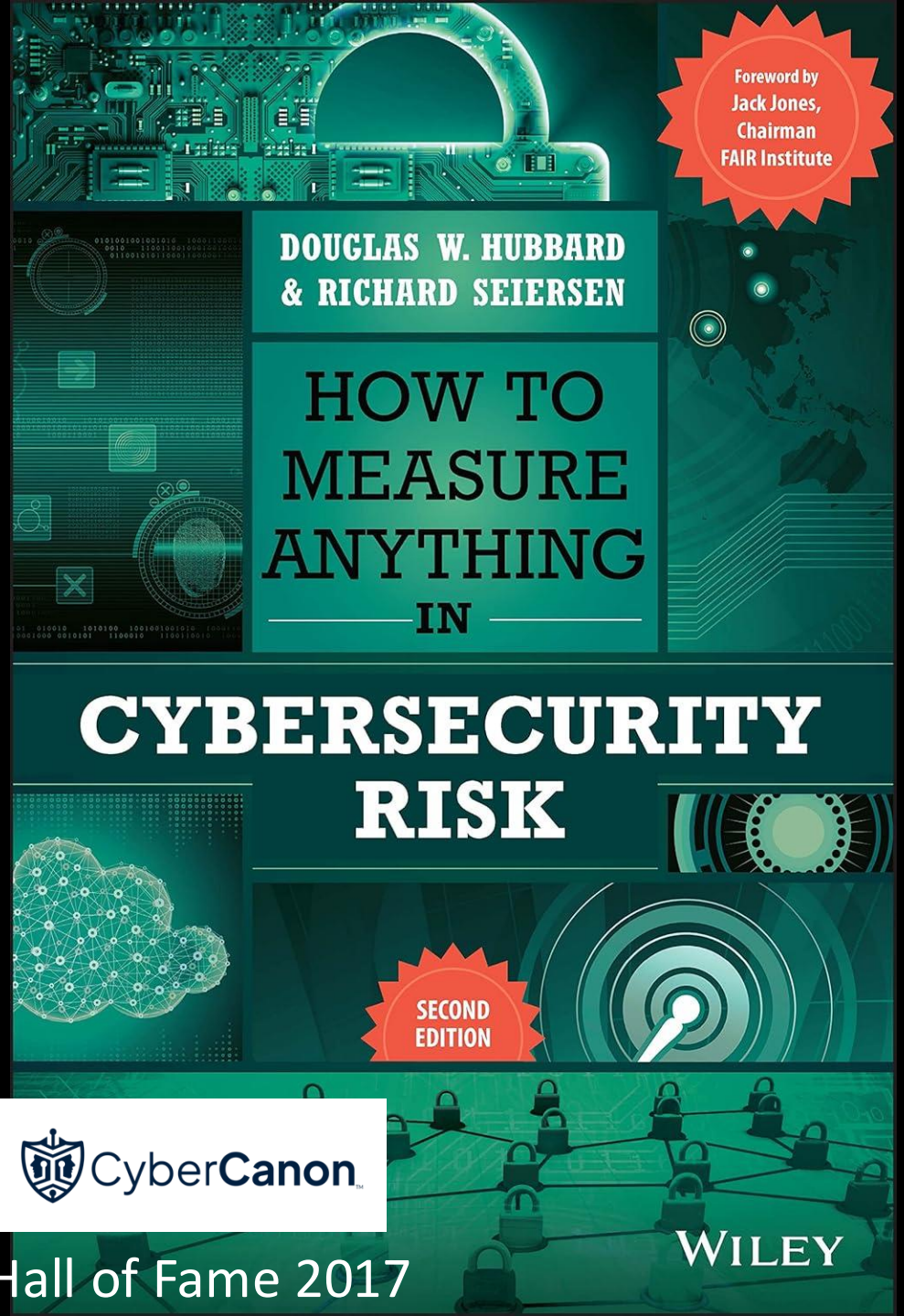
The Risk Matrix

Bold cells = “Critical” (score ≥ 20).
B lands in the Critical corner.
A lands one cell below.

	I=1	I=2	I=3	I=4	I=5
L=5	5	10	15	20 ← **B**	25
L=4	4	8	12	16	20
L=3	3	6	9	12	15 ← **A**
L=2	2	4	6	8	10
L=1	1	2	3	4	5

Risk A’s expected loss is **~87× larger** than Risk B’s.
The matrix has the ranking reversed.

Risk	Actual likelihood	Actual impact	Expected annual loss
A (breach)	15% (top of bucket 3)	**$40M**	**$6,000,000**
B (BEC)	60% (bottom of bucket 5)	$115K	$69,000



CyberCanon

Hall of Fame 2017

6 Truths of Cyber Risk Quanti

Ask Gemini

philvenables.com/post/6-truths-of-cyber-risk-quantification

New Chrome available

RISK & CYBERSECURITY

Thoughts from the Field

HOME ABOUT EVENTS AND PUBLICATIONS

All Posts Leadership Risk Cybersecurity Technology

Phil Venables · Sep 7, 2024 · 8 min read

6 Truths of Cyber Risk Quantification

I wrote the original version of this post over 4 years ago. In revisiting this it is interesting to note that not much has actually advanced in the field. Yes, there have been more products and tools developed to apply FAIR or FAIR-like quantitative methods - some successful and some less so, usually indexed on the degree of effort it takes to set up the tooling to get more value out than you put in.

As with other areas of risk there's a Heisenberg-like quality to much of the approaches. That is the act of measuring often changes the situation, often positively. Although the *Breaking Bad* use of the term Heisenberg might also apply given the mix of confusion and euphoria that often results from excess use of risk quantification.

There have also been significant advances in the world of cyber-risk insurance particularly in insurers fine tuning their actuarial models, having more data available from the insured parties, and from industry sources like research teams, intelligence units and some cyber scoring services. Some cloud providers and cybersecurity companies have also partnered to stimulate capacity and provide risk-adjusted pricing.

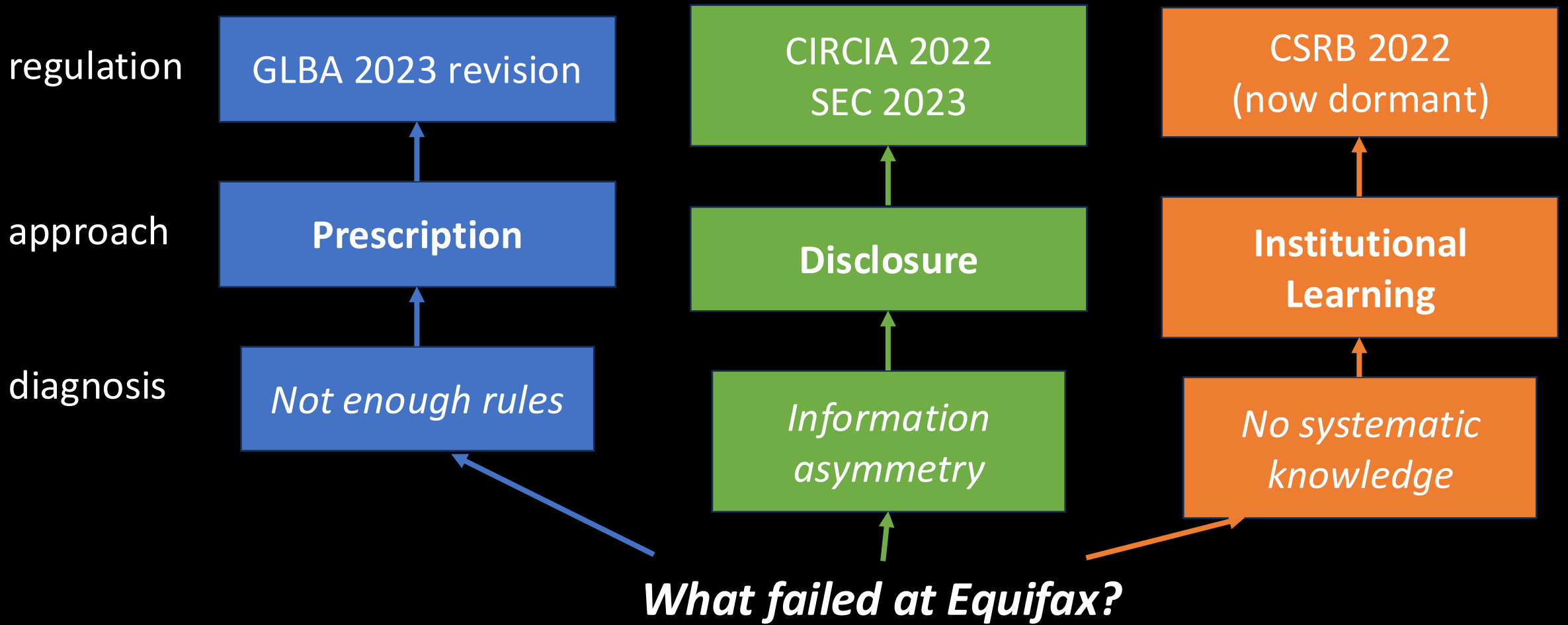
But overall, I think there has been the most progress in the effective use of data, analytics

Why This Is Hard

- **Nobody wants to commit to numbers** — executives push back on probabilities
- Teams **argue about estimates** — but that’s the point! It surfaces hidden disagreements
- **Risk registers become compliance artifacts** — quantitative models used to justify decisions already made

- Idea: **Forecasting** to build up strength and trust in estimates

Regulation as a Guardrail



Compliance Theater vs. Useful Improvements

FEDERAL TRADE COMMISSION

16 CFR Part 314

RIN 3084–AB35

Standards for Safeguarding Customer Information

AGENCY: Federal Trade Commission.

ACTION: Final rule.

SUMMARY: The Federal Trade Commission ("FTC" or "Commission") is issuing a final rule ("Final Rule") to amend the Standards for Safeguarding Customer Information ("Safeguards Rule" or "Rule"). The Final Rule contains five main modifications to the existing Rule. First, it adds provisions designed to provide covered financial institutions with more guidance on how to develop and implement specific aspects of an overall information security program, such as access controls, authentication, and encryption. Second, it adds provisions designed to improve the accountability of financial institutions' information security programs, such as by requiring periodic reports to boards of directors or governing bodies. Third, it exempts financial institutions that collect less customer information from certain requirements. Fourth, it expands the definition of "financial institution" to include entities engaged in activities the Federal Reserve Board determines to be incidental to financial activities. This change adds "finders"—companies that bring together buyers and sellers of a product or service—within the scope of the Rule. Finally, the Final Rule defines several terms and provides related examples in the Rule itself rather than incorporates them from the Privacy of Consumer Financial Information Rule ("Privacy Rule").

DATES:

Effective date: This rule is effective January 10, 2022.

Applicability date: The provisions set forth in § 314.5 are applicable beginning December 9, 2022.

FOR FURTHER INFORMATION CONTACT:

The GLBA provides a framework for regulating the privacy and data security practices of a broad range of financial institutions. Among other things, the GLBA requires financial institutions to provide customers with information about the institutions' privacy practices and about their opt-out rights, and to implement security safeguards for customer information.

Subtitle A of Title V of the GLBA required the Commission and other Federal agencies to establish standards for financial institutions relating to administrative, technical, and physical safeguards for certain information.[2] Pursuant to the Act's directive, the Commission promulgated the Safeguards Rule (16 CFR part 314) in 2002. The Safeguards Rule became effective on May 23, 2003.

The current Safeguards Rule requires a financial institution to develop, implement, and maintain a comprehensive information security program that consists of the administrative, technical, and physical safeguards the financial institution uses to access, collect, distribute, process, protect, store, use, transmit, dispose of, or otherwise handle customer information.[3] The information security program must be written in one or more readily accessible parts.[4] The safeguards set forth in the program must be appropriate to the size and complexity of the financial institution, the nature and scope of its activities, and the sensitivity of any customer information at issue.[5] The safeguards must also be reasonably designed to ensure the security and confidentiality of customer information, protect against any anticipated threats or hazards to the security or integrity of the information, and protect against unauthorized access to or use of such information that could result in substantial harm or inconvenience to any customer.[6]

In order to develop, implement, and maintain its information security program, a financial institution must identify reasonably foreseeable internal assessment, and must regularly test or otherwise monitor the effectiveness of the safeguards' key controls, systems, and procedures.[8] The Rule also requires the financial institution to evaluate and adjust its information security program in light of the results of this testing and monitoring, any material changes in its operations or business arrangements, or any other circumstances it knows or has reason to know may have a material impact on its information security program.[9] The financial institution must also designate an employee or employees to coordinate the information security program.[10]

Finally, the current Safeguards Rule requires financial institutions to take reasonable steps to select and retain service providers capable of maintaining appropriate safeguards for customer information and require those service providers by contract to implement and maintain such safeguards.[11]

II. Regulatory Review of the Safeguards Rule

On September 7, 2016, the Commission solicited comments on the Safeguards Rule as part of its periodic review of its rules and guides.[12] The Commission sought comment on a number of general issues, including the economic impact and benefits of the Rule; possible conflicts between the Rule and state, local, or other Federal laws or regulations; and the effect on the Rule of any technological, economic, or other industry changes. The Commission received 28 comments from individuals and entities representing a wide range of viewpoints.[13] Most commenters agreed there is a continuing need for the Rule and it benefits consumers and competition.[14]

On April 4, 2019, the Commission issued a notice of proposed rulemaking (NPRM) setting forth proposed amendments to the Safeguards Rule (the "Proposed Rule").[15] In response, the Commission received 49 comments from various interested parties

The Compliance Theatre: Wh

Ask Gemini

cybersecurityadvisors.network/2025-12-02-the-compliance-theatre-when-red-t...

CYBERSECURITY ADVISORS NETWORK

HOME | WELCOME TO CYAN | BLOG | EVENTS | MENTORSHIP | MEMBERS DIRECTORY | JOIN CYAN

02/12/2025

THE COMPLIANCE THEATRE: WHEN RED TAPE MEETS CYBERSECURITY BY NICK KELLY

Title: "Compliance Achieved: Racoons in the bins", by Nick Kelly

THE SUFFOCATING EMBRACE OF ACCUMULATED LAW

The Government (I speak of the US Government in this article, although the principle argument is as good as a blueprint for many other governments globally) has developed a peculiar affliction over the past half-century: the inability to throw anything away. Rather like a hoarder whose home has become impassable due to accumulated newspapers and defunct appliances, modern government has layered law upon law, regulation upon regulation, until the original floor is no longer visible and movement has become nearly impossible. The Interstate Highway Act of 1956 ran to 29 pages and delivered the entire system in roughly 15 years. The Affordable Care Act of 2010 sprawled across 2,700 pages and its implementation remains contentious more than a decade later. One might observe a certain inverse relationship between page count and efficacy.

Philip K. Howard, lawyer and founder of Common Good, has spent decades documenting this phenomenon with the enthusiasm of a forensic archaeologist examining societal decay. In a recent appearance on The Economist's podcast, Howard articulated the fundamental problem with characteristic clarity: government requires spring cleaning. Not the superficial tidying that involves moving problems from one cupboard to another, nor the 'taking a chainsaw approach' of the short-lived DOGE (an utter catastrophe in this author's opinion, with dire societal consequences – see the uprooting of USAID leading to likely thousands of deaths, the culling of staff in Cybersecurity and Infrastructure Security Agency leading to a weakening of a key national security agency in the states, etc.), but a strategic decluttering that requires acknowledging that most of what we've accumulated no longer serves any useful purpose and should be consigned to the skip.

The mechanism of dysfunction is straightforward. Each crisis, each scandal, each failure prompts the addition of new requirements designed to prevent that specific failure from recurring. No one removes the old requirements, which were themselves responses to previous failures. And whilst the premise that ushers in d faith, the result is what Howard describes as:

https://cybersecurityadvisors.network/media/

STATE OF (IN)SECURITY NEWSLETTER

Recent Digests

CYBER (IN)SECURITIES – ISSUE 197

CYBER (IN)SECURITIES – ISSUE 196

CYBER (IN)SECURITIES – ISSUE 195

CYBER (IN)SECURITIES – ISSUE 194

CYBER (IN)SECURITIES – ISSUE 193

Explore More

View All CyAN Media Channels →

PARTNERS

CVE OF THE WEEK BY WHITE HAT IT SECURITY

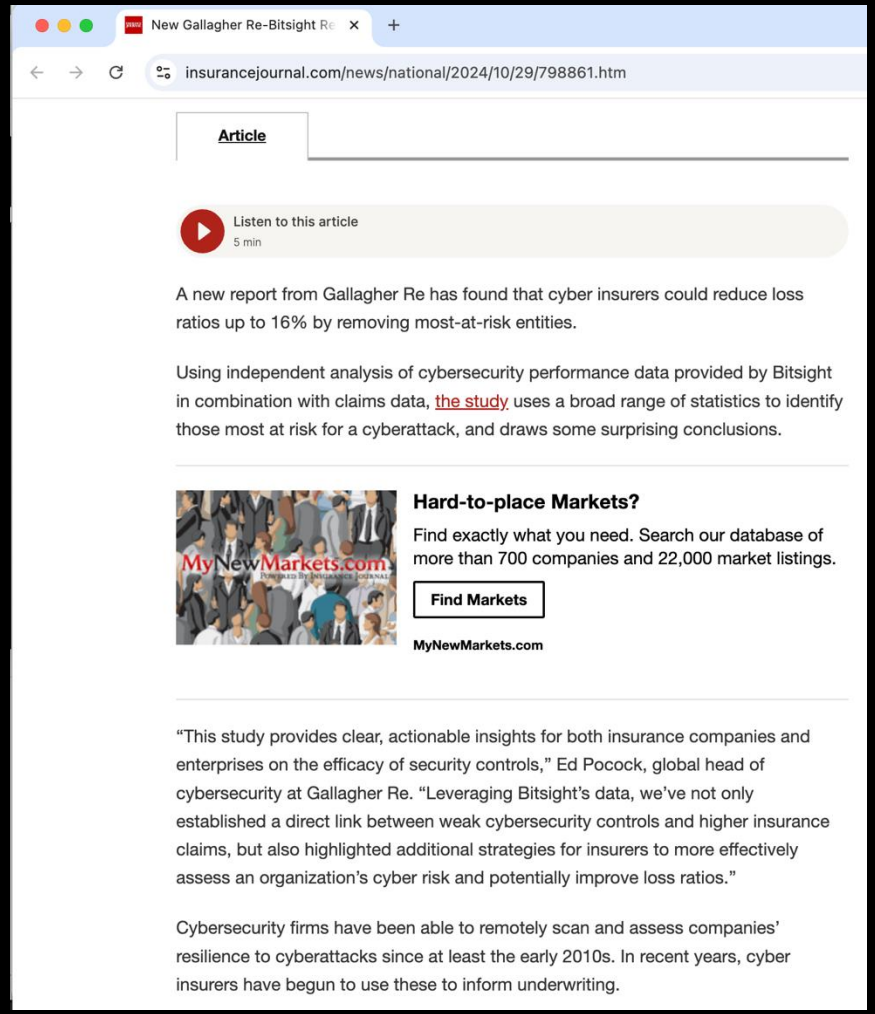
WEEK 16 – TRUSTED FORMAT, HIDDEN THREAT: EXPLOITING ADOBE READER VIA PDF

WEEK 15 – ONE TEXT AWAY: THE SAMSUNG EXYNOS ZERO-CLICK THREAT

WEEK 14 – CRACKED OPEN: A CRITICAL F5 FLAW HIDING INSIDE THE EASTER EGG

Upcoming Events

Cyber insurance: Elevating evidence?



New Gallagher Re-Bitsight Re

insurancejournal.com/news/national/2024/10/29/798861.htm

Article

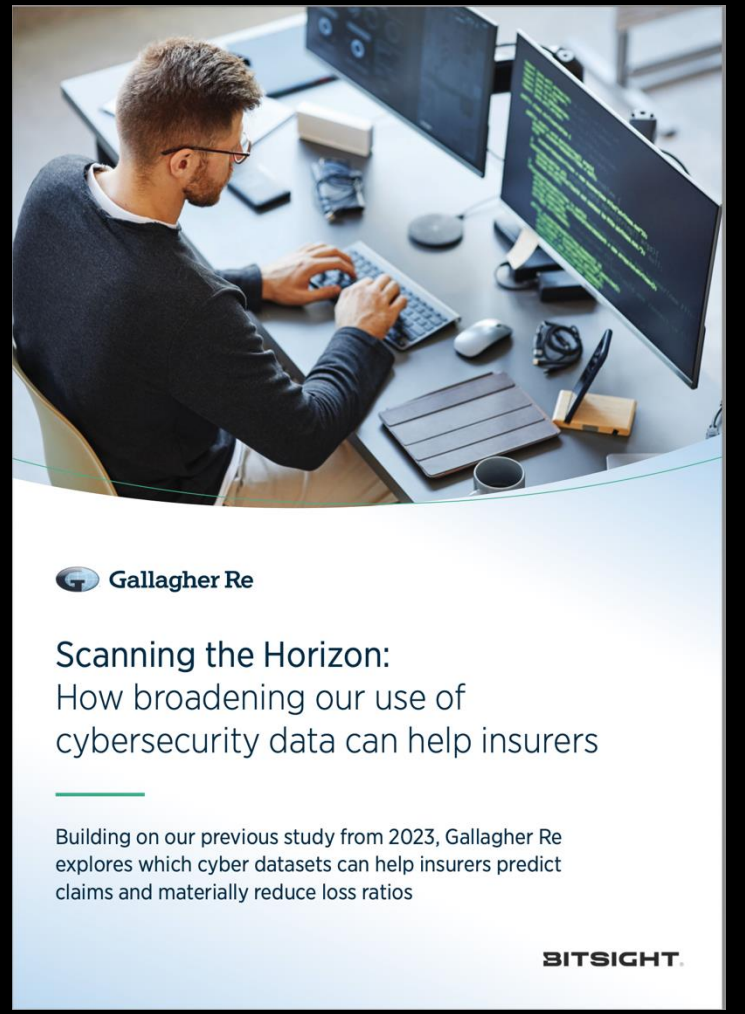
Listen to this article
5 min

A new report from Gallagher Re has found that cyber insurers could reduce loss ratios up to 16% by removing most-at-risk entities.

Using independent analysis of cybersecurity performance data provided by Bitsight in combination with claims data, the study uses a broad range of statistics to identify those most at risk for a cyberattack, and draws some surprising conclusions.

"This study provides clear, actionable insights for both insurance companies and enterprises on the efficacy of security controls," Ed Pocock, global head of cybersecurity at Gallagher Re. "Leveraging Bitsight's data, we've not only established a direct link between weak cybersecurity controls and higher insurance claims, but also highlighted additional strategies for insurers to more effectively assess an organization's cyber risk and potentially improve loss ratios."

Cybersecurity firms have been able to remotely scan and assess companies' resilience to cyberattacks since at least the early 2010s. In recent years, cyber insurers have begun to use these to inform underwriting.



Risk: Correlation is not Causation

- Simple regression (blue line): more security implies more losses?!
- Problem: Confounding variables (especially threat level)

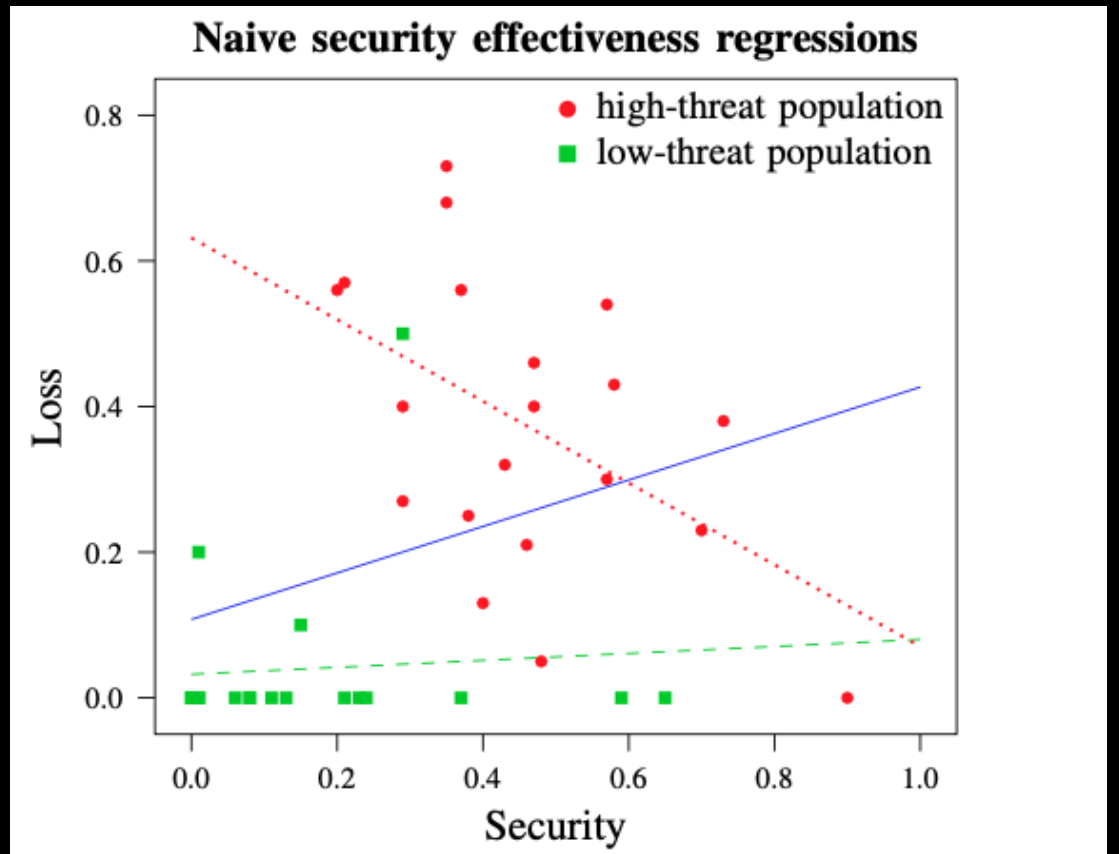


Fig. 1. The solid blue line fails to account for threat level, which may lead the high-threat population to under estimate the effectiveness of security.

SoK: Quantifying Cyber Risk

Daniel W. Woods
University of Innsbruck
Innsbruck, Austria

Rainer Böhme
University of Innsbruck
Innsbruck, Austria

Course goals: You will be able to

- **Understand cybersecurity from a data-driven and economic perspective**, learning to make decisions based on empirical evidence, following good science
- **Identify key vulnerabilities and threats**, especially when considering the **impact of humans**, both when they are attack targets and when the play a role in ensuring a system's security
- **Follow a well-designed process for secure systems construction**, from threat modeling to building to testing to maintenance
- **Manage security operations** – preventing, detecting, mitigating, and recovering from incidents – and gather data to improve future posture
- **Make risk-informed decisions**: Assess designs and technologies according to how they mitigate security risk, while leveraging **insurance** and responding to **regulation**
- **Communicate effectively and with empathy** to key stakeholders about security options and recommendations

All while taking a data-informed approach

Know Your Goal

> *"If the audience remembers only one thing from your talk, what should it be?"*
>
> — Simon Peyton Jones

Know Your Audience

A talk aimed at **everyone** reaches **no one**.

Customize ruthlessly.

The Overall Structure

Give them a roadmap, then take them on the journey

1. **Opening Hook** (1–2 min): Why should they care *right now*?
2. **Problem & Solution Teaser** (2–3 min): What's at stake & your recommendation
3. **Problem in Depth** (5–8 min): Evidence the problem is real & urgent
4. **Solution in Depth** (10–15 min): How it works, with examples
5. **Evidence / Results** (3–5 min): Data, case studies, demos
6. **Call to Action & Summary** (2–3 min): What you want them to do

For Security Specifically

- Lead with **business impact**, not technical details
- Use **data** when you have it; be honest about uncertainty when you don't
- Present **stackable options**: "We could do A, or A+B, or A+B+C"
- Reframe: **"This investment protects $X million in revenue"** not **"We might get hacked"**

Too Many Points EXAMPLE

5 malware variants flagged

Bangalore saw alerts → Minneapolis escalated to → No action

Auto-delete was off

Three points crammed into one slide. Everything is too small to read.

PART 4

Delivery: Bringing It to Life

No substitute for practice

Foundations | Structure | Slides | **Delivery** | Security | Process

Frame as Opportunity, Not FUD

Fear

"We might get hacked like Target."
"Bad things could happen."
"We're at risk."

Opportunity

"A $2M investment protects our $50M in card transactions."
"This reduces our exposure by ~80%."
"This puts us ahead of PCI-DSS 3.0 requirements."

Building Trust

You're building **trust**, not winning arguments.

- **Present trade-offs fairly,** even when you have a clear recommendation
- Be **transparent** about limitations and uncertainties
- **Trust your audience's judgment** — give options, let them choose

What CISOs Present to Boards

Surveyed Fortune 1000 CISOs on their actual board updates

All include:

- Changes to the **risk landscape**
- **Maturity scores**
- **Security initiatives** progress
- **Significant incidents**

- Changes to the **risk landscape**
- **Maturity scores**
- **Security initiatives** progress
- **Significant incidents**

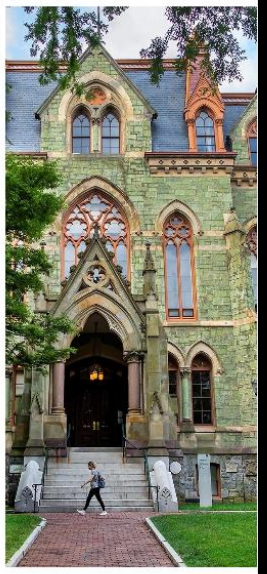
Board reporting

- **Typically once or twice per year in person, with a written update for other quarters**
- **Semi-standardized approach:**
  - **Top risks and updates on how they are being addressed**
  - **Project updates for major efforts**
  - **Metrics – Penn wide and focused on top ten applications**
  - **Review of high impact incidents**

23

University-Wide Information Security Metrics

Metric	Previous	Current
Patching (Target: less than 30 days) Measure adjusted to quarterly vs annual.	12.3 days average (Q1 FY26)	13.6 days average (Q1 FY26)
Serious Incident Detection Time (Target: less than 3 days)	30 Days (2 serious events)	NA (0 serious events)
Phishing Simulation Outcomes (Target: less than 15%)	32% (Fiscal YTD)	28% (Fiscal YTD)
Security Budget as a Percent of IT Budget	xxx (FY24)	xxx (FY25)



Security Risks & Projects Update

Risks	Actions	Action Status
Identities and Accounts	IAM Project	Removed hundreds of thousands of stale accounts from Kite Active Directory. Upgraded password reset support implemented for most PennKeys.
Decentralization	Penn SecureIT	Penn SecureIT project to consolidate SEVP IT functions in progress.
Ransomware	ISC CrowdStrike Complete pilot	Pilot success; expanding to SEVP centers as part of the above effort.
State Sponsored Hackers		
<location specific>	<improvement plan>	<progress>
Program Maturity/Scale	Accepted risk	See Penn SecureIT slides
Unmanaged Devices	Accepted risk	Accepted risk

Now that you've taken this course:

As an engineer, manager, or CISO:
What would you do to make your code and/or systems more secure?