Are We Skillful or Just Lucky?

Interpreting the Possible Histories of Vulnerability Disclosures

ALLEN D. HOUSEHOLDER and JONATHAN SPRING, CERT/CC®, Software Engineering Institute, Carnegie Mellon University

Coordinated Vulnerability Disclosure (CVD) stands as a consensus response to the persistent fact of vulnerable software, yet few performance indicators have been proposed to measure its efficacy at the broadest scales. In this article, we seek to fill that gap. We begin by deriving a model of all possible CVD histories from first principles, organizing those histories into a partial ordering based on a set of desired criteria. We then compute a baseline expectation for the frequency of each desired criteria and propose a new set of performance indicators to measure the efficacy of CVD practices based on the differentiation of skill and luck in observation data. As a proof of concept, we apply these indicators to a variety of longitudinal observations of CVD practice and find evidence of significant skill to be prevalent. We conclude with reflections on how this model and its accompanying performance indicators could be used by various stakeholders (vendors, system owners, coordinators, and governments) to interpret the quality of their CVD practices.

CCS Concepts: • **Security and privacy** → **Vulnerability management**; *Social aspects of security and privacy*;

Additional Key Words and Phrases: Vulnerability disclosure, vulnerability management, evaluation, measurement, process modelling, CSIRTs

An extended version of this article is available [21].

This material is based upon work funded and supported by the Department of Homeland Security under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center sponsored by the United States Department of Defense. The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation. NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT. [DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution. Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works. External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

* These restrictions do not apply to U.S. government entities. DM21-0018.

Authors’ address: A. D. Householder and J. Spring, CERT/CC®, Software Engineering Institute, Carnegie Mellon University, 4500 Fifth Ave, Pittsburgh, PA 15213; emails: {adh, jspring}@cert.org.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2022 Association for Computing Machinery.

2576-5337/2022/12-ART39 \$15.00

<https://doi.org/10.1145/3477431>

ACM Reference format:

Allen D. Householder and Jonathan Spring, 2022. Are We Skillful or Just Lucky? Interpreting the Possible Histories of Vulnerability Disclosures. *Digit. Threat.: Res. Pract.* 3, 4, Article 39 (December 2022), 28 pages.
<https://doi.org/10.1145/3477431>

1 INTRODUCTION

The practice of **Coordinated Vulnerability Disclosure (CVD)** emerged as part of a growing consensus to develop normative behaviors in response to the persistent fact of vulnerable software. Yet while the basic principles of Coordinated Vulnerability Disclosure (CVD) have been established [13, 22, 23, 30], to date there has been limited work to measure the efficacy of CVD programs, especially at the scale of industry benchmarks.

ISO 29147 [23] sets out the goals of vulnerability disclosure:

- (a) ensuring that identified vulnerabilities are addressed;
- (b) minimizing the risk from vulnerabilities;
- (c) providing users with sufficient information to evaluate risks from vulnerabilities to their systems;
- (d) setting expectations to promote positive communication and coordination among involved parties.

Meanwhile, use of third party libraries and shared code components across vendors and their products creates a need to coordinate across those parties whenever a vulnerability is found in a shared component. **Multi-Party Coordinated Vulnerability Disclosure (MPCVD)** is a more complex form of CVD, as illustrated by the Senate hearings about the Meltdown and Spectre vulnerabilities [31]. The need for Multi-Party Coordinated Vulnerability Disclosure (MPCVD) arises from the complexities of the software supply chain [18]. Nevertheless, the goals of CVD apply to MPCVD, as the latter is merely a special case of the former.

The difficulty of MPCVD derives from the diversity of its stakeholders: Software vendors have different development budgets, schedules, tempos, and analysis capabilities to do any of isolate, understand, or fix vulnerabilities. Additionally, they face diverse customer support expectations and obligations, and an increasing variety of regulatory regimes governing some stakeholders but not others. For these and many other reasons, practitioners of MPCVD highlight *fairness* as a core difficulty in coordinating disclosures across vendors [22].

So with the goal of minimizing the societal harm that results from the existence of a vulnerability in multiple products spread across multiple vendors, our motivating question is “What does *fair* mean in MPCVD?”. Optimizing MPCVD directly is not currently possible as we lack a utility function to map from the events that occur in a given case to the impact that case has on the world. While this article does not fully address the problem, it sets out a number of steps toward a solution. We seek a way to sort MPCVD cases into better or worse outcomes. Ideally the sorting criteria should be agreed based on unambiguous principles, which are intelligible by all interested parties. Furthermore, we seek a way to measure relevant features across MPCVD cases. Feature observability is a key factor: Our measurement needs to be simple and repeatable without relying overly much on proprietary or easily hidden information.

While a definition of *fair* in MPCVD is a responsibility for the broader community, we focus on evaluating the skill of the coordinator. We expect this contributes to fairness based on the EthicsFIRST principles of ethics for incident response teams promoted by **Forum of Incident Response and Security Teams (FIRST)** [39].¹ To that end, our research questions are:

- RQ1** : Construct a model of CVD states amenable to analysis and also future generalization to MPCVD.
- RQ2** : What is a reasonable baseline expectation for ordering of events in the model of CVD?
- RQ3** : Given this baseline and model, does CVD as observed “in the wild” demonstrate skillful behavior?

¹Specifically, skill in our model will align with fulfilling the duty of coordinated vulnerability disclosure, duty of confidentiality, duty to inform, duty to team ability, and duty of evidence-based reasoning.

This article primarily focuses on the simpler case of CVD. This focus provides an opportunity for incremental analysis of the success of the model; MPCVD modeling can follow in future work.

1.1 Contributions

The contributions of this article are as follows:

- We define a simple yet comprehensive model of possible disclosure histories in Section 3 and a set of criteria to order them in Section 4, which will address **RQ1**.
- We explore the implications of our model with respect to expected outcomes in Section 5, which will address **RQ2**.
- We propose a method for measuring the relative contribution of both skill and luck to observations of CVD outcomes over time in Section 6.
- We demonstrate the application of these techniques to analyze the efficacy of observed CVD processes in Section 7, which will address **RQ3**.

A discussion of how the model could be applied to benchmarks and multiparty coordination follows in Section 8. Section 9 describes the limitations of the approach and lays out future work to improve it. Section 10 surveys related work, and Section 11 summarizes and concludes.

2 EVENTS IN A VULNERABILITY LIFECYCLE

The goal of this section is to establish a model of events that affect the outcomes of vulnerability disclosure.

Our model builds on previous models of the vulnerability lifecycle, specifically those of Arbaugh et al. [1], Frei et al. [19], and Bilge and et al. [8]. A more thorough literature review of vulnerability lifecycle models can be found in [27].

Because we are modeling only the disclosure process, we assume the vulnerability both exists and is known to at least someone. Therefore, we ignore the *birth* (*creation*, *introduced*) and *discovery* states as they are implied at the beginning of all possible vulnerability disclosure histories. We also omit the *anti-virus signatures released* event from [8] since we are not attempting to model vulnerability management operations in detail.

The first event we are interested in modeling is Vendor Awareness (*V*). This event corresponds to *Disclosure* in [1] and *vulnerability discovered by vendor* in [8] (this event is not modeled in [19]). We are not concerned with *how* the vendor came to find out about the vulnerability's existence, whether it was found via internal testing, reported by a security researcher, or noticed as the result of incident analysis.

The second event we include is Public Awareness (*P*) of the vulnerability. This event corresponds to *Publication* in [1], *time of public disclosure* in [19], and *vulnerability disclosed publicly* in [8]. The public might find out about a vulnerability through the vendor's announcement of a fix, a news report about a security breach, a conference presentation by a researcher, by comparing released software versions as in [40, 41], or any of a variety of other means. As above, we are primarily concerned with the occurrence of the event itself rather than the details of *how* the *P* event arises.

The third event we address is Fix Readiness (*F*), by which we refer to the vendor's creation and possession of a fix that *could* be deployed to a vulnerable system, *if* the system owner knew of its existence. Here we differ somewhat from [1, 8, 19] in that their models address the *release* of the fix rather than its *readiness* for release.

The reason for this distinction will be made clear, but first we must mention that Fix deployed (*D*) is simply that: the fix exists, and it has been deployed.

We chose to include the Fix Ready (*F*), Fix Deployed (*D*), and Public Awareness (*P*) events so that our model could better accommodate two common modes of modern software deployment:

- *shrinkwrap* —The traditional distribution mode in which the vendor and deployer are distinct entities and deployers must be made aware of the fix before it can be deployed. In this case, which corresponds to the

previously mentioned *fix release* event, both fix readiness (F), and public awareness (P) are necessary for the fix to be deployed (D).

- *SaaS*—A more recent delivery mode in which the vendor also plays the role of deployer. In this distribution mode, fix readiness (F) can lead directly to fix deployed (D) with no dependency on public awareness (P).

We note that so-called *silent fixes* by vendors can sometimes result in a fix being deployed without public awareness even if the vendor is not the deployer. Thus, it is possible, albeit unlikely, for D to occur before P even in the *shrinkwrap* case above. It is also possible, and somewhat more likely, for P to occur before D in the *SaaS* case as well.

We diverge from [1, 8, 19] again in our treatment of exploits and attacks. Because attacks and exploit availability are often discretely observable events, the broader concept of *exploit automation* in [1] is insufficiently precise for our use. Both [8, 19] focus on the availability of exploits rather than attacks, but the observability of their chosen events is hampered by attackers' incentives to maintain stealth. Frei et al. [19] use *exploit availability*, whereas Bilge et al. [8] call it *exploit released in wild*. Both refer to the state in which an exploit is known to exist, but this can arise for at least two distinct reasons, which we wish to discriminate:

- *exploit public* (X)—when the method of exploitation for a vulnerability has been made public in sufficient detail to be reproduced by others. **Proof of concept (POC)** code posted to a widely available site or inclusion of the exploit in a commonly available exploit tool meets this criteria, whereas privately held exploits do not.
- *attacks observed* (A)—when the vulnerability has been observed to be exploited in attacks. In this case, one has evidence that the vulnerability has been exploited and can presume the existence of an exploit regardless of its availability to the public. Analysis of malware from an incident might meet A but not X depending on how closely held the malware is thought to be to the attacker. Use of an already public exploit in an attack meets both X and A .

Therefore, while we appreciate the existence of a hidden *exploit exists* event as causal predecessor of both *exploit public* and *attacks*, for our model we assert no causal relationship between X and A . We make this choice in the interest of observability. The *exploit exists* event is difficult to consistently observe independently of the two events we have chosen to use; its occurrence is nearly always inferred from the observation of either *exploit public* or *attacks*.

A summary of this model comparison is shown in Table 1. Further discussion of related work can be found in Section 10.

2.1 Definitions and Notation

Before we discuss either possible histories (Section 3) or desirable histories (Section 4) in the vulnerability life cycle, we need to formally define our terms. In all these definitions, we take standard Zermelo–Fraenkel set theory. The concept of sequences extends set theory to include a concept of ordered sets. From them, we adopt the following notation:

- $\{ \dots \}$ An unordered set, which makes no assertions about sequence.
- The normal proper subset ($\subset$), equality ($=$), and subset ($\subseteq$) relations between sets.
- $(\dots)$ An ordered set in which the events e occur in that sequence.
- The precedes ($<$) relation on members of an ordered set: $e_i < e_j$ if and only if $e_i, e_j \in s$ and $i < j$ where s is as defined in (2).

From Table 1, we define the set of events E

$$E \stackrel{\text{def}}{=} \{V, F, D, P, X, A\}. \quad (1)$$

Table 1. Vulnerability Lifecycle Events: Comparing Models

Arbaugh et al. [1]	Frei et al.[19]	Bilge et al.[8]	Our Model
Birth	creation (t_{creat})	introduced (t_c)	(implied)
Discovery	discovery (t_{disco})	n/a	(implied)
Disclosure	n/a	discovered by vendor (t_d)	Vendor Awareness (V)
n/a	patch availability (t_{patch})	n/a	Fix Ready (F)
Fix Release	n/a	patch released (t_p)	F and P
Publication	public disclosure (t_{discl})	disclosed publicly (t_0)	Public Awareness (P)
n/a	patch installation (t_{insta})	patch deployment completed (t_a)	Fix Deployed (D)
Exploit Automation	exploit availability (t_{explo})	Exploit released in wild (t_e)	n/a
Exploit Automation	n/a	n/a	Exploit Public (X)
Exploit Automation	n/a	n/a	Attacks (A)
n/a	n/a	anti-virus signatures released (t_s)	n/a

A sequence s is an ordered set of some number of events $e_i \in E$ for $1 \leq i \leq n$ and the length of s is $|s| \stackrel{\text{def}}{=} n$.

$$s \stackrel{\text{def}}{=} (e_1, e_2, \dots, e_n). \quad (2)$$

A valid vulnerability coordination history h is a sequence s containing one and only one of each of the event types in E ; by definition $|h| = |E| = 6$. Note this is a slight abuse of notation; $| \cdot |$ represents both sequence length and the cardinality of a set

$$h \stackrel{\text{def}}{=} s : \forall e_i, e_j \in s \text{ it is the case that } e_i \neq e_j \text{ and } \forall e_k \in E \text{ it is the case that } \exists e_i \in s \text{ such that } e_k = e_i, \quad (3)$$

where two members of the set E are equal if they are represented by the same symbol and not equal otherwise. The set of all possible histories, S_H , is a set of all the sequences h that satisfy this definition.

3 THE POSSIBLE HISTORIES OF CVD

Given that a history h contains all six events E in some order, there are at most 720 ($6P_6 = 6! = 720$) possible histories. That is, $|S_H| = 720$. However, we can apply causal constraints as follows:

- *vendor awareness* must precede *fix ready* ($V < F$);
- *fix ready* must precede *fix deployed* ($F < D$).

In symbols, this puts the following constraints on the possible set of histories, which we call H_0

$$H_0 \stackrel{\text{def}}{=} \{h \in S_H \text{ such that } \forall e \in h \text{ it is the case that } V < F \text{ and } F < D\}. \quad (4)$$

We further impose two simplifying assumptions. The first is that vendors know at least as much as the public does. In other words, all histories must meet one of two criteria: either *Vendor Awareness* V precedes *Public Awareness* P or else *Vendor Awareness* must immediately follow it

$$H_1 \stackrel{\text{def}}{=} \{h \in S_H \text{ such that } \forall e \in h \text{ it is the case that if } e_i = V \text{ then either } e_i < P \text{ or } e_{i-1} = P\}. \quad (5)$$

The second is that the public can be informed about a vulnerability by a public exploit. Therefore, either *public awareness* precedes *exploit public* or must immediately follow it

$$H_2 \stackrel{\text{def}}{=} \{h \in S_H \text{ such that } \forall e \in h \text{ it is the case that if } e_i = P \text{ then either } e_i < X \text{ or } e_{i-1} = X\}. \quad (6)$$

Combining them, we arrive at our formal definition of valid possible histories as all sequences meeting the three constraining assumptions (4), (5), and (6)

$$H \stackrel{\text{def}}{=} H_0 \cap H_1 \cap H_2. \quad (7)$$

Once these constraints are applied, only 70 possible histories $h \in S_H$ remain viable ($|H| = 70$). This model is amenable for analysis of CVD, but we need to add a way to express preferences before it is complete. Thus, we are part way through **RQ1**. Section 8.2 will address how this model can generalize from CVD to MPCVD.

These histories are listed exhaustively in Table 2. The skill ranking function on the histories will be defined in Section 5.5. The desirability of the history ($\mathbb{D}^h$) will be defined in Section 4. The expected frequency of each history f_h is explained in Section 5.2.

4 ON THE DESIRABILITY OF POSSIBLE HISTORIES

All histories are not equally preferable. Some are quite bad—for example, those in which attacks precede vendor awareness ($A < V$)—while others are very desirable, for example, those in which fixes are deployed before either an exploit is made public ($D < X$) or attacks ($D < A$).

In pursuit of a way to reason about our preferences for some histories over others, we define the following preference criteria: history h_a is preferred over history h_b if, all else being equal, a more desirable event e_1 precedes a less desirable event e_2 . In notation, $e_1 < e_2$. We define the following ordering preferences:

- $V < P$, $V < X$, or $V < A$ – Vendors can take no action to produce a fix if they are unaware of the vulnerability. Public awareness prior to vendor awareness can cause increased support costs for vendors at the same time they are experiencing increased pressure to prepare a fix. If public awareness of the vulnerability prior to vendor awareness is bad, then a public exploit is at least as bad because it encompasses the former and makes it readily evident that adversaries have exploit code available for use. Attacks prior to vendor awareness represent a complete failure of the vulnerability remediation process because they indicate that adversaries are far ahead of defenders.
- $F < P$, $F < X$, or $F < A$ – As noted above, the public can take no action until a fix is ready. Because public awareness also implies adversary awareness, the vendor/adversary race becomes even more critical if this condition is unmet. When fixes exist before exploits or attacks, defenders are better able to protect their users.
- $D < P$, $D < X$, or $D < A$ – Even better than vendor awareness and fix availability prior to public awareness, exploit publication, or attacks are scenarios in which fixes are deployed prior to one or more of those events.
- $P < X$ or $P < A$ – In many cases, D requires system owners to take action. We therefore prefer histories in which public awareness happens prior to either exploit publication or attacks.
- $X < A$ – This criteria is not about whether exploits should be published or not.² It is about whether we should prefer histories in which exploits are published *and then* attacks happen over histories in which attacks happen *and then* an exploit is published. Our position is that attackers have more advantages in the latter case than the former, and therefore we should prefer histories in which $X < A$.

²Although we do believe there is some value in exploit publication because it allows defenders to develop detection controls (e.g., in the form of behavioral patterns or signatures). Even if those detection mechanisms are imperfect, it seems better that they be in place prior to adversaries using them than the opposite.

Table 2. Possible Histories $h \in H$ of CVD

#	h	rank	$ \mathbb{D}^h $	f_h	$D < A$	$D < P$	$D < X$	$F < A$	$F < P$	$F < X$	$P < A$	$P < X$	$V < A$	$V < P$	$V < X$	$X < A$
0	AXPVFD	1	0	0.0833	0	0	0	0	0	0	0	0	0	0	0	0
1	APVXFD	2	2	0.0417	0	0	0	0	0	0	0	1	0	0	1	0
2	AVXPFD	3	2	0.0278	0	0	0	0	0	0	0	0	0	1	1	0
3	XPVAFD	4	3	0.1250	0	0	0	0	0	0	1	0	1	0	0	1
4	VAXPFD	5	3	0.0208	0	0	0	0	0	0	0	0	1	1	1	0
5	PVAXFD	6	4	0.0417	0	0	0	0	0	0	1	1	1	0	1	0
6	AVPXF	7	3	0.0139	0	0	0	0	0	0	0	1	0	1	1	0
7	APVFXD	7	3	0.0208	0	0	0	0	0	1	0	1	0	0	1	0
8	XPVFA	8	4	0.0625	0	0	0	1	0	0	1	0	1	0	0	1
9	VAPXFD	9	4	0.0104	0	0	0	0	0	0	0	1	1	1	1	0
10	PVXAFD	10	5	0.0417	0	0	0	0	0	0	1	1	1	0	1	1
11	VPAXFD	11	5	0.0104	0	0	0	0	0	0	1	1	1	1	1	0
12	PVAFXD	11	5	0.0208	0	0	0	0	0	1	1	1	1	0	1	0
13	VXPAFD	11	5	0.0312	0	0	0	0	0	0	1	0	1	1	1	1
14	AVPFXD	12	4	0.0069	0	0	0	0	0	1	0	1	0	1	1	0
15	APVFDX	13	4	0.0208	0	0	1	0	0	1	0	1	0	0	1	0
16	VAPFXD	14	5	0.0052	0	0	0	0	0	1	0	1	1	1	1	0
17	XPVFDA	15	5	0.0625	1	0	0	1	0	0	1	0	1	0	0	1
18	PVXFAD	16	6	0.0208	0	0	0	1	0	0	1	1	1	0	1	1
19	AVFXPD	17	4	0.0093	0	0	0	0	1	1	0	0	0	1	1	0
20	VPXAFD	18	6	0.0104	0	0	0	0	0	0	1	1	1	1	1	1
21	PVFAXD	19	6	0.0139	0	0	0	1	0	1	1	1	1	0	1	0
22	VXPFAD	19	6	0.0156	0	0	0	1	0	0	1	0	1	1	1	1
23	VPAFXD	20	6	0.0052	0	0	0	0	0	1	1	1	1	1	1	0
24	VAFXPD	21	5	0.0069	0	0	0	0	1	1	0	0	1	1	1	0
25	PVAFDX	22	6	0.0208	0	0	1	0	0	1	1	1	1	0	1	0
26	AVPFDX	23	5	0.0069	0	0	1	0	0	1	0	1	0	1	1	0
27	AVFPXD	24	5	0.0046	0	0	0	0	1	1	0	1	0	1	1	0
28	PVFXAD	25	7	0.0139	0	0	0	1	0	1	1	1	1	0	1	1
29	VPXFAD	25	7	0.0052	0	0	0	1	0	0	1	1	1	1	1	1
30	VAPFDX	26	6	0.0052	0	0	1	0	0	1	0	1	1	1	1	0
31	VAFPXD	27	6	0.0035	0	0	0	0	1	1	0	1	1	1	1	0
32	PVXFDA	28	7	0.0208	1	0	0	1	0	0	1	1	1	0	1	1
33	VPFAXD	29	7	0.0035	0	0	0	1	0	1	1	1	1	1	1	0
34	VFAXP	30	6	0.0052	0	0	0	1	1	1	0	0	1	1	1	0
35	VXPFDA	31	7	0.0156	1	0	0	1	0	0	1	0	1	1	1	1
36	PVFADX	32	7	0.0139	0	0	1	1	0	1	1	1	1	0	1	0
37	VPAFDX	33	7	0.0052	0	0	1	0	0	1	1	1	1	1	1	0
38	VPFXAD	34	8	0.0035	0	0	0	1	0	1	1	1	1	1	1	1
39	AVFPDX	35	6	0.0046	0	0	1	0	1	1	0	1	0	1	1	0
40	VFAPXD	36	7	0.0026	0	0	0	1	1	1	0	1	1	1	1	0
41	VPXFDA	37	8	0.0052	1	0	0	1	0	0	1	1	1	1	1	1
42	PVFXDA	37	8	0.0139	1	0	0	1	0	1	1	1	1	0	1	1
43	VAFPDX	38	7	0.0035	0	0	1	0	1	1	0	1	1	1	1	0
44	VPFADX	39	8	0.0035	0	0	1	1	0	1	1	1	1	1	1	0

(Continued)

Table 2. Continued

#	h	rank	$ \mathbb{D}^h $	f_h	$D < A$	$D < P$	$D < X$	$F < A$	$F < P$	$F < X$	$P < A$	$P < X$	$V < A$	$V < P$	$V < X$	$X < A$
45	VFPAXD	40	8	0.0026	0	0	0	1	1	1	1	1	1	1	1	0
46	VFXPAD	41	8	0.0078	0	0	0	1	1	1	1	0	1	1	1	1
47	AVFDXP	42	6	0.0046	0	1	1	0	1	1	0	0	0	1	1	0
48	PVFDAX	43	8	0.0139	1	0	1	1	0	1	1	1	1	0	1	0
49	VAFDXP	44	7	0.0035	0	1	1	0	1	1	0	0	1	1	1	0
50	VPFXDA	45	9	0.0035	1	0	0	1	0	1	1	1	1	1	1	1
51	VFAPDX	46	8	0.0026	0	0	1	1	1	1	0	1	1	1	1	0
52	VFPXAD	46	9	0.0026	0	0	0	1	1	1	1	1	1	1	1	1
53	AVFDPX	47	7	0.0046	0	1	1	0	1	1	0	1	0	1	1	0
54	PVFDXA	48	9	0.0139	1	0	1	1	0	1	1	1	1	0	1	1
55	VPFDAX	49	9	0.0035	1	0	1	1	0	1	1	1	1	1	1	0
56	VFXPDA	50	9	0.0078	1	0	0	1	1	1	1	0	1	1	1	1
57	VFPADX	51	9	0.0026	0	0	1	1	1	1	1	1	1	1	1	0
58	VAFDPX	52	8	0.0035	0	1	1	0	1	1	0	1	1	1	1	0
59	VFADXP	53	8	0.0026	0	1	1	1	1	1	0	0	1	1	1	0
60	VPFDXA	54	10	0.0035	1	0	1	1	0	1	1	1	1	1	1	1
61	VFPXDA	55	10	0.0026	1	0	0	1	1	1	1	1	1	1	1	1
62	VFADPX	56	9	0.0026	0	1	1	1	1	1	0	1	1	1	1	0
63	VFPDAX	57	10	0.0026	1	0	1	1	1	1	1	1	1	1	1	0
64	VFDAXP	58	9	0.0026	1	1	1	1	1	1	0	0	1	1	1	0
65	VFPDXA	59	11	0.0026	1	0	1	1	1	1	1	1	1	1	1	1
66	VFDAPX	60	10	0.0026	1	1	1	1	1	1	0	1	1	1	1	0
67	VFDXPA	61	11	0.0052	1	1	1	1	1	1	1	0	1	1	1	1
68	VFDPA X	61	11	0.0026	1	1	1	1	1	1	1	1	1	1	1	0
69	VFD PXA	62	12	0.0026	1	1	1	1	1	1	1	1	1	1	1	1

Equation (8) formalizes our definition of desired orderings $\mathbb{D}$. Table 3 displays all 36 possible orderings of paired events and whether they are considered impossible, required (as defined by Equation (4)), desirable (as defined by Equation (8)), or undesirable (the complement of the set defined in Equation (8)).

Before proceeding, we note that our model focuses on the ordering of events, not their timing. We acknowledge that in some situations, the interval between events may be of more interest than merely the order of those events, as a rapid tempo of events can alter the options available to stakeholders in their response. We discuss this limitation further in Section 9; however, the following model posits event sequence timing on a human-oriented timescale measured in minutes to weeks.

$$\begin{aligned}
\mathbb{D} \stackrel{\text{def}}{=} \{ & V < P, V < X, V < A, \\
& F < P, F < X, F < A, \\
& D < P, D < X, D < A, \\
& P < X, P < A, X < A \}.
\end{aligned} \tag{8}$$

An element $d \in \mathbb{D}$ is of the form $e_i < e_j$. More formally, d is a relation of the form $d(e_1, e_2, <)$. $\mathbb{D}$ is a set of such relations.

Given the desired preferences over orderings of events ($\mathbb{D}$ in Equation (8)), we can construct a partial ordering over all possible histories H , as defined in Equation (10). This partial order requires a formal definition of which

Table 3. Ordered Pairs of Events

	V	F	D	P	X	A
V	-	r	r	d	d	d
F	-	-	r	d	d	d
D	-	-	-	d	d	d
P	u	u	u	-	d	d
X	u	u	u	u	-	d
A	u	u	u	u	u	-

where $row < col$ (Key: - = impossible, r = required, d = desired, and u = undesired).

desiderata are met by a given history, provided by (9).

$$\begin{aligned} \mathbb{D}^h &\stackrel{\text{def}}{=} \{d \in \mathbb{D} \text{ such that } d \text{ is true for } h\}, \text{ for } h \in H \\ &\text{where } d(e_1, e_2, <) \text{ is true for } h \text{ if and only if:} \\ \exists e_i, e_j \in h \text{ such that } e_i = e_1 \text{ and } e_j = e_2 \text{ and } h \text{ satisfies the relation } d(e_i, e_j, <). \end{aligned} \quad (9)$$

$$(H, \leq_H) \stackrel{\text{def}}{=} \forall h_a, h_b \in H \text{ it is the case that } h_b \leq_H h_a \text{ if and only if } \mathbb{D}^{h_b} \subseteq \mathbb{D}^{h_a}. \quad (10)$$

A visualization of the resulting partially ordered set, or poset, $(H, \leq_H)$ is shown as a Hasse Diagram in Figure 1. Hasse Diagrams represent the transitive reduction of a poset. Each node in the diagram represents an individual history h_a from Table 2; labels correspond to the index of the table. Figure 1 follows Equation (10), in that h_a is higher in the order than h_b when h_a contains all the desiderata from h_b and at least one more. Histories that do not share a path are incomparable (formally, two histories incomparable if both $\mathbb{D}^{h_a} \not\supseteq \mathbb{D}^{h_b}$ and $\mathbb{D}^{h_b} \not\supseteq \mathbb{D}^{h_a}$). The diagram flows from least desirable histories at the bottom to most desirable at the top. This model satisfies **RQ1**; Sections 5 and 6 will demonstrate that the model is amenable to analysis and Section 8.2 will lay out the criteria for extending it to cover MPCVD.

The poset $(H, \leq_H)$, has as its upper bound $h_{69} = (V, F, D, P, X, A)$, while its lower bound is $h_0 = (A, X, P, V, F, D)$.

Thus far, we have made no assertions about the relative desirability of any two desiderata (that is, $d_i, d_j \in \mathbb{D}$ where $i \neq j$). In the next section, we will expand the model to include a partial order over our desiderata, but for now it is sufficient to note that any simple ordering over $\mathbb{D}$ would remain compatible with the partial order given in Equation (10). In fact, a total order on $\mathbb{D}$ would create a linear extension of the poset defined here, whereas a partial order on $\mathbb{D}$ would result in a more constrained poset of which this poset would be a subset.

5 REASONING OVER POSSIBLE HISTORIES

Our goal in this section is to formulate a way to rank our undifferentiated desiderata $\mathbb{D}$ from Section 4 in order to develop the concept of CVD skill and its measurement in Section 6. This will provide a baseline expectation about events (**RQ2**).

In order to begin to differentiate skill from chance in Section 6, we need a model of what the CVD world would look like without any skill. We cannot derive this model by observation. Even when CVD was first practiced in the 1980s, some people may have had social, technical, or organizational skills that transferred to better CVD. We follow the principle of indifference, as stated in [17]:

Principle of Indifference: Let $X = \{x_1, x_2, \dots, x_n\}$ be a partition of the set W of possible worlds into n mutually exclusive and jointly exhaustive possibilities. In the absence of any relevant evidence

Fig. 1. The Lattice of Possible CVD Histories: A Hasse Diagram of the partial ordering $(H, \leq_H)$ of $h_a \in H$ given $\mathbb{D}$ as defined in Equation (10). The diagram flows from least desirable histories at the bottom to most desirable at the top. Histories that do not share a path are incomparable. Labels indicate the index (row number) a of h_a in Table 2.

pertaining to which cell of the partition is the true one, a rational agent should assign an equal initial credence of n to each cell.

While the principle of indifference is rather strong, it is inherently a bit difficult to reason about absolutely skill-less CVD when the work of CVD is, by its nature, a skilled job. We will use the principle of indifference to define a baseline against which measurement can be meaningful. For additional analysis of the application of the principle of indifference to this problem, see [21, Section 3.3].

5.1 Event Frequency Analysis

We model event frequency with a simple state-based model of the possible histories $h \in H$ in which each state is a binary vector indicating which events $e \in E$ have occurred prior to reaching that state. The events $e \in E$ therefore represent state transitions, and the histories $h \in H$ are paths (traces) through the states. This meets the definition above because each $e \in E$ is unique (mutually exclusive) and the set of available e at each step of the way is exhaustive. Let E_{i+1}^h be the set of possible next events following the i th event in history h , which is a subset of all possible events: $E_{i+1}^h \subseteq E$. The fragment of a history h up to its i th element is a sequence, which contains the first i events of h , denoted as h_i . The initial case $h_0 \stackrel{\text{def}}{=} \emptyset$. The probability of transition from e_i to

any of the possible next events e_{i+1} , where $e_{i+1} \in E_{i+1}^h$, is defined, based on the principle of indifference, as the inverse of the cardinality of the set of possible states, namely:

$$p(e_{i+1}|h_i) = 1/|E_{i+1}^h|. \quad (11)$$

For example, because Equation (4) requires $V < F$ and $F < D$, only four of the six events in E are possible at the beginning of a history: $\{V, P, X, A\}$. Therefore, $p(F|\emptyset) = p(D|\emptyset) = 0$. Since the principle of indifference assigns each possible transition event as equally probable in this model of unskilled CVD, we assign an initial probability of 0.25 to each possible event ($p(V|\emptyset) = p(P|\emptyset) = p(X|\emptyset) = p(A|\emptyset) = 0.25$). From there, we see that the other rules dictate possible transitions from each subsequent state. For example, Equation (5) says that any h starting with P must start with PV . And Equation (6) requires any h starting with X must proceed through XP and again Equation (5) gets us to XPV . Therefore, we expect histories starting with PV or XPV to occur with frequency 0.25 as well.

5.2 History Frequency Analysis

We apply the principle of indifference to the available events ($E_{i+1}^h \subseteq E$) at each state i for each of the possible histories to compute the expected frequency of each history, which we denote as f_h . The frequency of a history f_h is the cumulative product of the probability p of each event e in the history h . We are only concerned with histories that meet our sequence constraints, namely, $h \in H$

$$f_h = \prod_{i=0}^5 p(e_{i+1}|h_i). \quad (12)$$

Table 2 displays the value of f_h for each history. Having an expected frequency (f_h) for each history h will allow us to examine how often we might expect our desiderata $d \in \mathbb{D}$ to occur across H .

Choosing uniformly over event transitions is more useful than treating the six-element histories as uniformly distributed. For example, $P < A$ in 59% of valid histories, but when histories are weighted by the assumption of uniform state transitions $P < A$ is expected to occur in 67% of the time. These differences arise due to the dependencies between some states. Since CVD practice is comprised a sequence of events, each informed by the last, our uniform distribution over events is more likely a useful baseline than a uniform distribution over histories.

5.3 Event Order Frequency Analysis

Each of the event pair orderings in Table 3 can be treated as a Boolean condition that either holds or does not hold in any given history.

In Section 5.2, we described how to compute the expected frequency of each history (f_h) given the presumption of indifference to possible events at each step. We can use f_h as a weighting factor to compute the expected frequency of event orderings ($e_i < e_j$) across all possible histories H . Equations (13) and (14) define the frequency of an ordering $f_{e_i < e_j}$ as the sum over all histories in which the ordering occurs ($H^{e_i < e_j}$) of the frequency of each such history (f_h) as shown in Table 2.

$$H^{e_i < e_j} \stackrel{\text{def}}{=} \{h \in H \text{ where } e_i < e_j \text{ is true for } h \text{ and } i \neq j\}, \quad (13)$$

$$f_{e_i < e_j} \stackrel{\text{def}}{=} \sum_{h \in H^{e_i < e_j}} f_h. \quad (14)$$

Table 4 displays the results of this calculation. Required event orderings have an expected frequency of 1, while impossible orderings have an expected frequency of 0. As defined in Section 4, each desiderata $d \in \mathbb{D}$ is specified as an event ordering of the form $e_i < e_j$. We use f_d to denote the expected frequency of a given desiderata $d \in \mathbb{D}$.

Table 4. Expected Frequency of $row < col$ When Events are Chosen Uniformly from Possible Events at Each Point

	V	F	D	P	X	A
V	0	1	1	0.333	0.667	0.75
F	0	0	1	0.111	0.333	0.375
D	0	0	0	0.037	0.167	0.187
P	0.667	0.889	0.963	0	0.5	0.667
X	0.333	0.667	0.833	0.5	0	0.5
A	0.25	0.625	0.812	0.333	0.5	0

The values for the relevant f_d appear in the upper right of Table 4. Some event orderings have higher expected frequencies than others. For example, vendor awareness precedes attacks in 3 out of 4 histories in a uniform distribution of event transitions ($f_{V < A} = 0.75$), whereas fix deployed prior to public awareness holds in less than 1 out of 25 ($f_{D < P} = 0.037$) histories generated by a uniform distribution over event transitions.

5.4 A Partial Order on Desiderata

Any observations of phenomena in which we measure the performance of human actors can attribute some portion of the outcome to skill and some portion to chance [15, 26]. It is reasonable to wonder whether good outcomes in CVD are the result of luck or skill. How can we tell the difference?

We begin with a simple model in which outcomes o are a combination of luck and skill,

$$o_{observed} = o_{luck} + o_{skill}. \quad (15)$$

In other words, outcomes due to skill are what remain when you subtract the outcomes due to luck from the outcomes you observe. In this model, we treat *luck* as a random component: the contribution of chance. In a world where neither attackers nor defenders held any advantage and events were chosen uniformly from E whenever they were possible, we would expect to see the preferred orderings occur with probability equivalent to their frequency f_d as shown in Table 4.

Skill, on the other hand, accounts for the outcomes once luck has been accounted for. So the more likely an outcome is due to luck, the less skill we can infer when it is observed. As an example, from Table 4 we see that fix deployed before the vulnerability is public is the rarest of our desiderata with $f_{D < P} = 0.037$, and thus exhibits the most skill when observed. On the other hand, vendor awareness before attacks is expected to be a common occurrence with $f_{V < A} = 0.75$.

We can therefore use the set of f_d to construct a partial order over $\mathbb{D}$ in which we prefer desiderata d , which are more rare (and therefore imply more skill when observed) over those that are more common. We create the partial order on $\mathbb{D}$ as follows: for any pair $d_1, d_2 \in \mathbb{D}$, we say that d_2 exhibits less skill than d_1 if d_2 occurs more frequently in H than d_1

$$(\mathbb{D}, \leq_{\mathbb{D}}) \stackrel{\text{def}}{=} d_2 \leq_{\mathbb{D}} d_1 \iff f_{d_2} \stackrel{\mathbb{R}}{\geq} f_{d_1}. \quad (16)$$

Note that the inequalities on the left and right sides of Equation (16) are flipped because skill is inversely proportional to luck. Also, while $\leq_{\mathbb{D}}$ on the left side of Equation (16) defines a preorder over the poset H , the $\stackrel{\mathbb{R}}{\geq}$ is the usual ordering over the set of real numbers. The result is a partial order $(\mathbb{D}, \leq_{\mathbb{D}})$ because a few d have the same f_d ($f_{F < X} = f_{V < P} = 0.333$, for example). The full Hasse Diagram for the partial order $(\mathbb{D}, \leq_{\mathbb{D}})$ is shown in Figure 2.

Fig. 2. Hasse Diagram of the partial order $(\mathbb{D}, \leq_{\mathbb{D}})$ defined in Equation (16) where the rarity of each d as shown in Table 4 is taken to reflect skill. Nodes at the top of the diagram reflect the most skill.

5.5 Ordering Possible Histories by Skill

Next we develop a new partial order on H given the partial order $(\mathbb{D}, \leq_{\mathbb{D}})$ just described. We observe that $\mathbb{D}^h$ acts as a Boolean vector of desiderata met by a given h . Since $0 \leq f_d \leq 1$, simply taking its inverse could in the general case lead to some large values for rare events, so for convenience we use $\log(1/f_d)$ as our proxy for skill. Taking the dot product of $\mathbb{D}^h$ with the set of $\log(1/f_d)$ represented as a vector, we arrive at a single value representing the skill exhibited for each history h . Careful readers may note that this value is equivalent to the **Term Frequency—Inverse Document Frequency (TF-IDF)** score for a search for the “skill terms” represented by $\mathbb{D}$ across the corpus of possible histories H .

We have now computed a skill value for every $h \in H$, which allows us to sort H and assign a rank to each history h contained therein. The rank is shown in Table 2. Rank values start at 1 for least skill up to a maximum of 62 for most skill. Owing to the partial order $(\mathbb{D}, \leq_{\mathbb{D}})$, some h have the same computed skill values, and these are given the same rank.

The ranks for $h \in H$ lead directly to a new poset $(H, \leq_{\mathbb{D}})$, which is an extension of and fully compatible with $(H, \leq_H)$ as developed in Section 4. The resulting Hasse Diagram would be too large to reproduce here. Instead, we include the resulting rank for each h as a column in Table 2. In the table, rank is ordered from least desirable

and skillful histories to most. Histories having identical rank are incomparable to each other within the poset. The refined poset $(H, \leq_{\mathbb{D}})$ is much closer to a total order on H , as indicated by the relatively few histories having duplicate ranks.

The remaining incomparable histories are the direct result of the incomparable d in $(\mathbb{D}, \leq_{\mathbb{D}})$, corresponding to the branches in Figure 2. Achieving a total order on $\mathbb{D}$ would require answering the following: Assuming you could achieve only one, and without regard to any other $d \in \mathbb{D}$, would you prefer

- that fix ready precede exploit publication ($F < X$) or that vendor awareness precede public awareness ($V < P$)?
- that public awareness precede exploit publication ($P < X$) or that exploit publication precede attacks ($X < A$)?
- that public awareness precede attacks ($P < A$) or vendor awareness precede exploit publication ($V < X$)?

Recognizing that readers may have diverse opinions on all three questions, we leave further analysis of the answers to these as future work.

This is just one example of how poset refinements might be used to order H . Different posets on $\mathbb{D}$ would lead to different posets on H . For example, one might construct a different poset if certain d were considered to have much higher financial value when achieved than others.

6 DISCRIMINATING SKILL AND LUCK IN OBSERVATIONS

This section defines a method for measuring skillful behavior in CVD, which we will need to answer **RQ3** about measuring and evaluating CVD “in the wild.” The measurement method makes use of all the modeling tools and baselines established thus far: A comprehensive set of possible histories H , a partial order over them in terms of the presence of desired event precedence $\mathbb{D}$, and the *a priori* expected frequency of each desiderata $d \in \mathbb{D}$.

If we expected to be able to observe all events in all CVD cases, we could be assured of having complete histories and could be done here. But the real world is messy. Not all events $e \in E$ are always observable. We need to develop a way to make sense of what we *can* observe, regardless of whether we are ever able to capture complete histories. Continuing towards our goal of measuring efficacy, we return to considering the balance between skill and luck in determining our observed outcomes.

Of course, there are any number of conceivable reasons why we should expect our observations to differ from the expected frequencies we established in Section 5. Adversaries might be rare, or conversely very well equipped. Vendors might be very good at releasing fixes faster than adversaries can discover vulnerabilities and develop exploits for them. System owners might be diligent at applying patches. We did say *might*, did we not? Regardless, for now we will lump all of those possible explanations into a single attribute we will call “skill.”

In a world of pure skill, one would expect that a player could achieve all 12 desiderata $d \in \mathbb{D}$ consistently. That is, a maximally skillful player could consistently achieve the specific ordering $h = (V, F, D, P, X, A)$ with probability $p_s = 1$.

Thus, we construct the following model: For each of our preferred orderings $d \in \mathbb{D}$, we model their occurrence due to luck using the binomial distribution with parameter $p_l = f_d$ taken from Table 4.

Recall that the mean of a binomial distribution is simply the probability of success p , and that the mean of a weighted mixture of two binomial distributions is simply the weighted mixture of the individual means. Therefore, our model adds a parameter α_d to represent the weighting between our success rates arising from skill p_s and luck p_l . Because there are 12 desiderata $d \in \mathbb{D}$, each d will have its own observations and corresponding value for α_d for each history h_a .

$$f_d^{obs} = \alpha_d p_s + (1 - \alpha_d) p_l. \quad (17)$$

Where f_d^{obs} is the observed frequency of successes for desiderata d . Because $p_s = 1$, one of those binomial distributions is degenerate. Substituting $p_s = 1$, $p_l = f_d$ and solving Equation (17) for α , we get

$$\alpha_d = \frac{f_d^{obs} - f_d}{1 - f_d}. \quad (18)$$

The value of α_d therefore gives us a measure of the observed skill normalized against the background success rate provided by luck f_d .

We denote the set of α_d values for a given history as $\alpha_{\mathbb{D}}$. When we refer to the α_d coefficient for a specific d we will use the specific ordering as the subscript, for example: $\alpha_{F < P}$

$$\alpha_{\mathbb{D}} = \{\alpha_d : d \in \mathbb{D}\}. \quad (19)$$

The concept embodied by f_d is founded on the idea that if attackers and defenders are in a state of equilibrium, the frequency of observed outcomes (i.e., how often each desiderata d and history h actually occurs) will appear consistent with those predicted by chance. So another way of interpreting α_d is as a measure of the degree to which a set of observed histories is out of equilibrium.

The following are a few comments on how α_d behaves. Note that $\alpha_d < 0$ when $0 \leq f_d^{obs} < f_d$ and $0 \leq \alpha_d \leq 1$ when $f_d \leq f_d^{obs} \leq 1$. The implication is that a negative value for α_d indicates that our observed outcomes are actually *worse* than those predicted by pure luck. In other words, we can only infer positive skill when the observations are higher ($f_d^{obs} > f_d$). That makes intuitive sense: If you are likely to win purely by chance, then you have to attribute most of your wins to luck rather than skill. The highest value for any $\mathbb{D}$ in Table 4 is $f_{V < A} = 0.75$, implying that even if a vendor only knows about 7 out of 10 vulnerabilities before attacks occur ($f_{V < A}^{obs} = 0.7$), they are still not doing better than random.

On the other hand, when f_d is small it is easier to infer skill should we observe anything better than f_d . However, it takes larger increments of observations f_d^{obs} to infer growth in skill when f_d is small than when it is large. The smallest f_d we see in Table 4 is $f_{D < P} = 0.037$.

Inherent to the binomial distribution is the expectation that the variance of results is lower for both extremes (as p approaches 0 or 1) and highest at $p = 0.5$. Therefore, we should generally be less certain of our observations when they fall in the middle of the distribution. We address uncertainty further in Section 7.2.

7 OBSERVING CVD IN THE WILD

As a proof of concept, in this section, we will demonstrate the use of the model developed in Section 6 to observational data. To begin, Section 7.1 refines the model to accommodate observational data. Next we address uncertainty in observations in Section 7.2. Finally, Sections 7.3 and 7.4 apply the model to two data sets: Microsoft's security updates from 2017 through early 2020, and commodity public exploits from 2015–2019.

7.1 Computing α_d from Observations

Although Equation (18) develops a skill metric from observed frequencies, our observations will in fact be based on counts. Observations consist of some number of successes S_d^o out of some number of trials T , i.e.,

$$f_d^{obs} = \frac{S_d^o}{T}. \quad (20)$$

We likewise revisit our interpretation of f_d

$$f_d = \frac{S_d^l}{T}, \quad (21)$$

where S_d^l is the number of successes at d we would expect due to luck in T trials.

Substituting Equations (20) and (21) into Equation (18), and recalling that $p_s = 1$ because a maximally skillful player succeeds in T out of T trials, we get

$$\alpha = \frac{\frac{S_d^o}{T} - \frac{S_d^l}{T}}{\frac{T}{T} - \frac{S_d^l}{T}}. \quad (22)$$

Rearranging Equation (21) to $S_d^l = f_d T$, substituting into Equation (22), and simplifying, we get

$$\alpha = \frac{S_d^o - f_d T}{(1 - f_d)T}. \quad (23)$$

Hence, for any of our desiderata $\mathbb{D}$ we can compute α_d given S_d^o observed successes out of T trials in light of f_d taken from Table 4.

Before we address the data analysis we take a moment to discuss uncertainty.

7.2 Calculating Measurement Error

We have already described the basis of our f_d^{obs} model in the binomial distribution. While we could just estimate the error in our observations using the binomial's variance $np(1-p)$, because of boundary conditions at 0 and 1 we should not assume symmetric error. An extensive discussion of uncertainty in the binomial distribution is given in [9].

However, for our purpose the Beta distribution lends itself to this problem nicely. The Beta distribution is specified by two parameters (a, b) . It is common to interpret a as the number of successes and b as the number of failures in a set of observations of Bernoulli trials to estimate the mean of the binomial distribution from which the observations are drawn. For any given mean, the width of the Beta distribution narrows as the total number of trials increases.

We use this interpretation to estimate a 95% credible interval for f_d^{obs} using a Beta distribution with parameters $a = S_d^o$ as successes and $b = T - S_d^o$ representing the number of failures using the `scipy.stats.beta.interval` function in Python. This gives us an upper and lower estimate for f_d^{obs} , which we multiply by T to get upper and lower estimates of S_d^o as in Equation (20).

7.3 Microsoft 2017–2020

We are now ready to proceed with our data analysis. First, we examine Microsoft's monthly security updates for the period between March 2017 and May 2020, as curated by the Zero Day Initiative (ZDI) blog.³ Figure 3(a) shows monthly totals for all vulnerabilities, while Figure 3(b) has monthly observations of $P < F$ and $A < F$. This dataset allowed us to compute the monthly counts for $F < P$ and $F < A$.

One benefit of this definition of possible CVD histories in Section 3 is an opportunity to clarify definitions of related terms. While not a focus of the analysis in this article, a brief comment on different interpretations of the term “zero day vulnerability” is helpful at this point. For example, one reviewer stated they prefer to define “zero day vulnerability” as $X < V$ and not $(P < F \text{ or } A < F)$. We should seek these precise definitions because sometimes both $X < V$ and $P < F$, in which case two people might agree that an instance is a zero day without understanding they disagree on the definition. We do not disagree with the reviewer that $X < V$ is a definition of zero day, it just is not the one we are using for this analysis, since $(P < F \text{ or } A < F)$ is what ZDI used as their definition. For example, perhaps $X < V$ is a zero day for a software vendor whereas $(P < F \text{ or } A < F)$ is a zero

³ <https://www.zerodayinitiative.com/blog>. The ZDI blog posts were more directly useful than the monthly Microsoft security updates because ZDI had already condensed the counts of how many vulnerabilities were known (P) or exploited (A) prior to their fix availability F . Retrieving this data from Microsoft's published vulnerability information requires collecting it from all the individual vulnerabilities patched each month. We are grateful to ZDI for providing this summary and saving us the effort.

Fig. 3. Publicly Disclosed Microsoft Vulnerabilities 2017–2020.

Fig. 4. Publicly Disclosed Microsoft Vulnerabilities 2017–2020.

day from the perspective of a system owner. An extended version of this article brings additional formalism to the various definitions of “zero day” [21]. However, for present purposes the ability to accurately and cleanly state our definition is enough.

Observations of $F < P$: In total, Microsoft issued patches for 2,694 vulnerabilities; 2,610 (0.97) of them met the fix-ready-before-public-awareness ($F < P$) objective. The mean monthly $\alpha_{F<P} = 0.967$, with a range of [0.878, 1.0]. We can also use the cumulative data to estimate an overall skill level for the observation period, which gives us a bit more precision on $\alpha_{F<P} = 0.969$ with the 0.95 interval of [0.962, 0.975]. Figure 4(a) shows the trend for both the monthly observations and the cumulative estimate of $\alpha_{F<P}$.

Observations of $F < A$: Meanwhile, 2,655 (0.99) vulnerabilities met the fix-ready-before-attacks-observed ($F < A$) criteria. Thus, we compute a mean monthly $\alpha_{F<A} = 0.976$ with range [0.893, 1.0]. The cumulative estimate yields $\alpha_{F<A} = 0.986$ with an interval of [0.980, 0.989]. The trend for both is shown in Figure 4(b).

Inferring Histories from Observations: Another possible application of our model is to estimate unobserved α_d based on the cumulative observations of both $f_{F<P}^{obs}$ and $f_{F<A}^{obs}$ above. Here, we estimate the frequency f_d of the other $d \in \mathbb{D}$ for this period. Our procedure is as follows:

- (1) For 10,000 rounds, draw an f_d^{est} for both $F < P$ and $F < A$ from the Beta distribution with parameters $a = S_d^o$ and $b = T - S_d^o$ where S_d^o is 2,610 or 2,655, respectively, and T is 2,694.
- (2) Assign each $h \in H$ a weight according to standard joint probability based whether it meets both, either, or neither $A = F < P$ and $B = F < A$, respectively.

Fig. 5. Simulated skill α_d for Microsoft 2017–2020 based on observations of $F < P$ and $F < A$ over the period.

$$w_h = \begin{cases} p_{AB} = f_A * f_B & \text{if } A \text{ and } B \\ p_{Ab} = f_A * f_b & \text{if } A \text{ and } \neg B \\ p_{aB} = f_a * f_B & \text{if } \neg A \text{ and } B \\ p_{ab} = f_a * f_b & \text{if } \neg A \text{ and } \neg B \end{cases}, \quad (24)$$

where $f_a = 1 - f_A$ and $f_b = 1 - f_B$

- (3) Draw a weighted sample (with replacement) of size $N = 2,694$ from H according to these weights.
- (4) Compute the sample frequency $f_d^{sample} = S_d^{sample}/N$ for each $d \in \mathbb{D}$, and record the median rank of all histories h in the sample.
- (5) Compute the estimated frequency as the mean of the sample frequencies $f_d^{est} = \overline{f_d^{sample}}$ for each $d \in \mathbb{D}$.
- (6) Compute α_d from f_d^{est} for each $d \in \mathbb{D}$.

As one might expect given the causal requirement that vendor awareness precedes fix availability, the estimated values of α_d are quite high (0.96–0.99) for our desiderata involving either V or F . We also estimate that α_d is positive—indicating that we are observing skill over and above mere luck—for all d except $P < A$ and $X < A$, which are slightly negative. The results are shown in Figure 5. The most common sample median history rank across all runs is 53, with all sample median history ranks falling between 51–55. The median rank of possible histories weighted according to the assumption of equiprobable transitions is 11. We take this as evidence that the observations are indicative of skill.

7.4 Commodity Exploits 2015–2019

Next, we examine the overall trend in $P < X$ for commodity exploits between 2015 and 2019. The dataset is based on the National Vulnerability Database [32], in conjunction with the CERT Vulnerability Data Archive [11]. Between these two databases, a number of candidate dates are available to represent the date a vulnerability was made public. We use the minimum of these as the date for P .

To estimate the exploit availability (X) date, we extracted the date a CVE-ID appeared in the git logs for Metasploit [36] or Exploitdb [33]. When multiple dates were available for a CVE-ID, we kept the earliest. Note that commodity exploit tools such as Metasploit and Exploitdb represent a non-random sample of the exploits available to adversaries. These observations should be taken as a lower bounds estimate of exploit availability, and therefore an upper bounds estimate of observed desiderata d and skill α_d .

During the time period from 2013–2019, the dataset contains 73,474 vulnerabilities. Of these, 1,186 were observed to have public exploits (X) prior to the earliest observed vulnerability disclosure date (P), giving an overall

Fig. 6. $\alpha_{P < X}$ for all NVD vulnerabilities 2013–2019 (X observations based on Metasploit and ExploitDb).

Fig. 7. Simulated skill α_d for all NVD vulnerabilities 2013–2019 based on observations of $P < X$ over the period.

success rate for $P < X$ of 0.984. The mean monthly $\alpha_{P < X}$ is 0.966 with a range of $[0.873, 1.0]$. The volatility of this measurement appears to be higher than that of the Microsoft data. The cumulative $\alpha_{P < X}$ comes in at 0.968 with an interval spanning $[0.966, 0.970]$. A chart of the trend is shown in Figure 6.

To estimate unobserved α_d from the commodity exploit observations, we repeat the procedure outlined in Section 7.3. This time, we use $N = 73,474$ and estimate f_d^{est} for $P < X$ with Beta parameters $a = 72,288$ and $b = 1186$. As above, we find evidence of skill in positive estimates of α_d for all desiderata except $P < A$ and $X < A$, which are negative. The most common sample median history rank in this estimate is 33 with a range of $[32, 33]$, which is lower than the median rank of 53 in the Microsoft estimate from Section 7.3, still beats the median rank of 11 assuming uniform event probabilities. The results are shown in Figure 7.

8 DISCUSSION

The observational analysis in Section 7 supports an affirmative response to **RQ3**: vulnerability disclosure as currently practiced demonstrates skill. In both datasets examined, our estimated α_d is positive for most $d \in \mathbb{D}$. However, there is uncertainty in our estimates due to the application of the principle of indifference to

unobserved data. This principle assumes a uniform distribution across event transitions in the absence of CVD, which is an assumption we cannot readily test. The spread of the estimates in Figures 5 and 7 represents the variance in our samples, not this assumption-based uncertainty. Our interpretation of α_d values near 0 is therefore that they reflect an absence of evidence rather than evidence that skill is absent. While we cannot rule definitively on luck or low skill, values of $\alpha_d > 0.9$ should reliably indicate skillful defenders.

If, as seems plausible from the evidence, it turns out that further observations of h are significantly skewed toward the higher end of the poset $(H, \leq_{\mathbb{D}})$, then it may be useful to empirically calibrate our metrics rather than using the *a priori* frequencies in Table 4 as our baseline. This analysis baseline would provide context on “more skillful than the average for some set of teams” rather than more skillful than blind luck. Section 8.1 discusses this topic, which should be viewed as an examination of what “reasonable” in **RQ2** should mean in the context of “reasonable baseline expectation.”

Section 8.2 suggests how the model might be applied to establish benchmarks for CVD processes involving any number of participants, which closes the analysis of **RQ1** in relation to MPCVD. Section 8.3 surveys the stakeholders in CVD and how they might use our model; The stakeholders are vendors, system owners, the security research community, coordinators, and governments. In particular, we focus on how these stakeholders might respond to the affirmative answer to **RQ3** and a method to measure skill in a way more tailored to each stakeholder group.

8.1 Benchmarks

As described above, in an ideal CVD situation, each observed history would achieve all 11 desiderata $\mathbb{D}$. Realistically, this is unlikely to happen. We can at least state that we would prefer that most cases reach fix ready before attacks ($F < A$). Per Table 4, even in a world without skill we would expect $F < A$ to hold in 73% of cases. This means that $\alpha_{F < A} < 0$ for anything less than a 0.73 success rate. Doing just barely better than random ($\epsilon > \alpha_d > 0$ for $\epsilon \approx 0$) is not terribly satisfying, so we would like to seek outcomes in which $\alpha_{F < A} \geq c_{F < A} \geq 0$ for some benchmark constant $c_{F < A}$. In fact, we propose to generalize this for any $d \in \mathbb{D}$, such that α_d should be greater than some benchmark constant c_d

$$\alpha_d \geq c_d \geq 0, \quad (25)$$

where c_d is based on observations of α_d collected across some collection of CVD cases.

We propose as a starting point a naïve benchmark of $c_d = 0$. This is a low bar, as it only requires that CVD actually do better than possible events which are independent and identically distributed (i.i.d.) within each case. For example, given a history in which (V, F, P) have already happened, D , X , or A are equally likely to occur next.

The i.i.d. assumption may not be warranted. We anticipate that event ordering probabilities might be conditional on history: for example, $p(X|P) > p(X|\neg P)$ or $p(A|X) > p(A|\neg X)$. If the i.i.d. assumption fails to hold for $e \in E$, observed frequencies of $h \in H$ could differ significantly from the rates predicted by the uniform probability assumption behind Table 4.

Some example suggestive observations are:

- There is reason to suspect that only a fraction of vulnerabilities ever reach the *exploit public* event X , and fewer still reach the *attack* event A . Recent work by the Cyentia Institute found that “5% of all CVEs are both observed within organizations AND known to be exploited”[14], which suggests that $f_{D < A} \approx 0.95$.
- Likewise, $D < X$ holds in 28 of 70 (0.4) h . However, Cyentia found that “15.6% of all open vulnerabilities observed across organizational assets in our sample have known exploits”[14], which suggests that $f_{D < X} \approx 0.844$.

On their own these observations can equally well support the idea that we are broadly observing skill in vulnerability response, rather than that the world is biased from some other cause. However, we could choose

a slightly different goal than differentiating skill and “blind luck” as represented by the i.i.d. assumption. One could aim at measuring “more skillful than the average for some set of teams” rather than more skillful than blind luck.

If this were the “reasonable” baseline expectation (**RQ2**), the primary limitation is available observations. This model helps overcome this limitation because it provides a clear path toward collecting relevant observations. For example, by collecting dates for the six $e \in E$ for a large sample of vulnerabilities, we can get better estimates of the relative frequency of each history h in the real world. It seems as though better data would serve more to improve benchmarks rather than change expectations of the role of chance.

As an applied example, if we take the first item in the above list as a broad observation of $f_{D < A} = 0.95$, we can plug into Equation (18) to get a potential benchmark of $\alpha_{D < A} = 0.94$, which is considerably higher than the naïve generic benchmark $\alpha_d = 0$. It also implies that we should expect actual observations of $h \in H$ to skew toward the 19 h in which $D < A$ around 19x as often as the 51 h in which $D \not< A$. Similarly, if we interpret the second item as a broad observation of $f_{D < X} = 0.844$, we can then compute a benchmark $\alpha_{D < X} = 0.81$, which is again a significant improvement over the naïve $\alpha_d = 0$ benchmark.

8.2 Applicability to MPCVD

MPCVD occurs when multiple vendors or stakeholders are involved in the disclosure process. The need for MPCVD arises due to the inherent nature of the software supply chain [22]. A vulnerability that affects a low-level component (such as a library or operating system API) can require fixes from both the originating vendor and any vendor whose products incorporate the affected component. Alternatively, vulnerabilities are sometimes found in protocol specifications or other design-time issues where multiple vendors may have each implemented their own components based on a vulnerable design.

A common problem in MPCVD is that of fairness: Coordinators are often motivated to optimize the CVD process to maximize the deployment of fixes to as many end users as possible while minimizing the exposure of users of other affected products to unnecessary risks.

The model presented in this article provides a way for coordinators to assess the effectiveness of their MPCVD cases. In an MPCVD case, each vendor/product pair effectively has its own 6-event history h_a . We can therefore recast MPCVD as a set of histories M drawn from the possible histories H :

$$M = \{h_1, h_2, \dots, h_m \text{ where each } h_a \in H\}. \quad (26)$$

Where $m = |M| \geq 1$. The edge case when $|M| = 1$ is simply the regular (non-multiparty) case.

We can then set desired criteria for the set M , as in the benchmarks described in Section 8.1. In the MPCVD case, we propose to generalize the benchmark concept such that the median $\tilde{\alpha}_d$ should be greater than some benchmark constant c_d

$$\tilde{\alpha}_d \geq c_d \geq 0. \quad (27)$$

In real-world cases where some outcomes across different vendor/product pairs will necessarily be lower than others, we can also add the criteria that we want the variance of each α_d to be low. An MPCVD case having high median α_d with low variance across vendors and products involved will mean that most vendors achieved acceptable outcomes.

To summarize:

- The median α_d for all histories $h \in M$ should be positive and preferably above some benchmark constant c_d , which may be different for each $d \in \mathbb{D}$.

$$\text{Median}(\{\alpha_d(h) : h \in M\}) \geq c_d > 0. \quad (28)$$

- The variance of each α_d for all histories $h \in M$ should be low. The constant ε is presumed to be small.

$$\sigma^2(\{\alpha_d(h) : h \in M\}) \leq \varepsilon. \quad (29)$$

Table 5. Ordering Preferences for Selected Stakeholders

$d \in \mathbb{D}$	Vendor ($\mathbb{D}_v$)	SysOwner ($\mathbb{D}_s$)	Coordinator ($\mathbb{D}_c$)
$V < P$	yes	maybe ⁴	yes
$V < X$	yes	maybe ⁴	yes
$V < A$	yes	maybe ⁴	yes
$F < P$	yes	maybe ⁵	yes
$F < X$	yes	yes	yes
$F < A$	yes	yes	yes
$D < P$	maybe ¹	maybe ¹	yes
$D < X$	maybe ²	maybe ⁵	yes
$D < A$	maybe ²	yes	yes
$P < X$	yes	yes	yes
$P < A$	yes	yes	yes
$X < A$	maybe ³	maybe ³	maybe ³

¹When vendors control deployment, both vendors and system owners likely prefer $D < P$. When system owners control deployment, $D < P$ is impossible.

²Vendors should care about orderings involving D when they control deployment, but might be less concerned if deployment responsibility is left to system owners.

³Exploit publication can be controversial. To some, it enables defenders to test deployed systems for vulnerabilities or detect attempted exploitation. To others, it provides unnecessary adversary advantage.

⁴System owners may only be concerned with orderings involving V insofar as it is a prerequisite for F .

⁵System owners might be indifferent to $F < P$ and $D < X$ depending on their risk tolerance.

8.3 Reflections on the Influence of CVD Roles

CVD stakeholders include vendors, system owners, the security research community, coordinators, and governments [22]. Different stakeholders might want different things, although most benevolent parties will likely seek some subset of $\mathbb{D}$. Because H is the same for all stakeholders, the expected frequencies shown in Table 4 will be consistent across any such variations in desiderata. A discussion of some stakeholder preferences is given below, while a summary can be found in Table 5. We notate these variations of the set of desiderata $\mathbb{D}$ with subscripts: $\mathbb{D}_v$ for vendors, $\mathbb{D}_s$ for system owners, $\mathbb{D}_c$ for coordinators, and $\mathbb{D}_g$ for governments. In Table 3, we defined a preference ordering between every possible pairing of events; therefore, $\mathbb{D}$ is the largest possible set of desiderata. We thus expect the desiderata of benevolent stakeholders to be a subset of $\mathbb{D}$ in most cases. That said, we note a few exceptions in the text that follows.

8.3.1 Vendors. As shown in Table 5, we expect vendors' desiderata $\mathbb{D}_v$ to be a subset of $\mathbb{D}$. It seems reasonable to expect vendors to prefer that a fix is ready before either exploit publication or attacks ($F < X$ and $F < A$, respectively). Fix availability implies vendor awareness ($V < F$), so we would expect vendors' desiderata to include those orderings as well ($V < X$ and $V < A$, respectively).

Vendors typically want to have a fix ready before the public finds out about a vulnerability ($F < P$). We surmise that a vendor's preference for this item could be driven by at least two factors: the vendor's tolerance for potentially increased support costs (e.g., fielding customer support calls while the fix is being prepared), and the perception that public awareness without an available fix leads to a higher risk of attacks.

When a vendor has control over fix deployment (D), it will likely prefer that deployment precede public awareness, exploit publication, and attacks ($D < P$, $D < X$, and $D < A$, respectively).⁴ However, when fix deployment depends on system owners to take action, the feasibility of $D < P$ is limited.⁵ Regardless of the vendor's ability to deploy fixes or influence their deployment, it would not be unreasonable for them to prefer that public awareness precedes both public exploits and attacks ($P < X$ and $P < A$, respectively).

Ensuring the ease of patch deployment by system owners remains a likely concern for vendors. Conscientious vendors might still prefer $D < X$ and $D < A$ even if they have no direct control over those factors. However, vendors may be indifferent to $X < A$.

Although our model only addresses event ordering, not timing, a few comments about timing of events are relevant here since they reflect the underlying process from which H arises. Vendors have significant influence over the speed of V to F based on their vulnerability handling, remediation, and development processes [24]. They can also influence how early V happens based on promoting a cooperative atmosphere with the security researcher community [23]. Vendor architecture and business decisions affect the speed of F to D . Cloud-based services and automated patch delivery can shorten the lag between F and D . Vendors that leave deployment contingent on system owner action can be expected to have longer lags, making it harder to achieve the $D < P$, $D < X$, and $D < A$ objectives, respectively.

8.3.2 System Owners. System owners ultimately determine the lag from F to D based on their processes for system inventory, scanning, prioritization, patch testing, and deployment—in other words, their **vulnerability management (VM)** practices. In cases where the vendor and system owner are distinct entities, system owners should optimize to minimize the lag between F and D in order to improve the chances of meeting the $D < X$ and $D < A$ objectives, respectively.

System owners might select a different desiderata subset than vendors $\mathbb{D}_s \subseteq \mathbb{D}$, $\mathbb{D}_s \neq \mathbb{D}_v$. In general, system owners are primarily concerned with the F and D events relative to X and A . Therefore, we expect system owners to be concerned about $F < X$, $F < A$, $D < X$, and $D < A$. As discussed above, $D < P$ is only possible when the vendor controls D . Depending on the system owner's risk tolerance, $F < P$ and $D < X$ may or may not be preferred. Some system owners may find $X < A$ useful for testing their infrastructure, others might prefer that no public exploits be available.

8.3.3 Security Researchers. The “friendly” offensive security community (i.e., those who research vulnerabilities, report them to vendors, and sometimes release proof-of-concept exploits for system security evaluation purposes) can do their part to ensure that vendors are aware of vulnerabilities as early as possible prior to public disclosure ($V < P$). They can also delay the publication of exploits until after fixes exist ($F < X$) and possibly even until most system owners have deployed the fix ($D < X$). This does not preclude adversaries from doing their own exploit development on the way to A , but it avoids providing them with unnecessary assistance.

8.3.4 Coordinators. Coordinators have been characterized as seeking to balance the social good across both vendors and system owners [7]. This implies that they are likely interested in the union of the vendors' and system owners' preferences. In other words, coordinators want the full set of desiderata ($\mathbb{D}_c = \mathbb{D}$).

We pause for a brief aside about the design of the model with respect to the coordination role. We considered adding a *Coordinator Awareness (C)* event, but this would expand $|H|$ from 70 to 452 because it could occur at any point in any h . There is not much for a coordinator to do once the fix is deployed; however, so we could

⁴On the other hand, it is possible that some vendors might actually prefer public awareness before fix deployment even if they have the ability to deploy fixes, for example, in support of transparency or trust building. In that case, $\mathbb{D}_v \not\subseteq \mathbb{D}$, and some portions of the analysis presented here may not apply.

⁵“Silent patches” can obviously occur when vendors fix a vulnerability but do not make that fact known. In principle, it is possible that silent patches could achieve $D < P$ even in traditional COTS or OSS distribution models. However, in practice silent patches result in poor deployment rates precisely because they lack an explicit imperative to deploy the fix.

potentially reduce $|H|$ to 329 by only including positions in H that precede the D event. This is still too large and unwieldy for meaningful analysis within our scope; instead, we simply provide the following comment.

The goal of coordination is this: regardless of which stage a coordinator becomes involved in a case, the objective is to choose actions that make preferred histories more likely and non-preferred histories less likely. A careful reading of the Hasse Diagram in Figure 1 or the ranking in Table 2 can suggest available actions to improve outcomes. Namely this means focusing coordination efforts as needed on vendor awareness, fix availability, fix deployment, and the appropriately timed public awareness of vulnerabilities and their exploits.

8.3.5 Governments. In their defensive roles, governments are essentially acting as a combination of system owners, vendors, and—increasingly—coordinators. Therefore, we might anticipate $\mathbb{D}_g = \mathbb{D}_c = \mathbb{D}$.

However, governments sometimes also have an adversarial role to play for national security, law enforcement, or other reasons. The model presented in this article could be adapted to that role as well by drawing some desiderata from the lower left triangle of Table 3. While defining such adversarial desiderata ($\mathbb{D}_a$) is out of scope for this article, we leave the topic with our expectation that $\mathbb{D}_a \not\subseteq \mathbb{D}$.

9 LIMITATIONS AND FUTURE WORK

This section highlights some limitations of the current work and lays out a path for improving on those limitations in future work. Broadly, the opportunities for expanding the model include modeling multiple agents, gathering more data about CVD in the world, working to account for fairness and MPCVD, addressing the importance of duration between events, options for modeling attacker behavior, and managing the impact of partial information.

Modeling Multiple Agents. We agree with the reviewer who suggested that an agent-based model could allow deeper examination of the interactions between stakeholders in MPCVD. Many of the mechanisms and proximate causes underlying the events this model describes are hidden from the model, and would be difficult to observe or measure even if they were included. Nevertheless, in order to reason about different stakeholders' strategies and approaches to MPCVD, we need a way to measure and compare outcomes. The model we present here gives us such a framework, but it does so by making a tradeoff in favor of generality over causal specificity. We anticipate that future agent-based models of MPCVD will be better positioned to address process mechanisms, whereas this model will be useful to assess outcomes independently of the mechanisms by which they arise.

Gather Data About CVD. Section 8.1 discusses how different benchmarks and “reasonable baseline expectations” might change the results of a skill assessment. It also proposes how to use observations of the actions a certain team or team performs to create a baseline, which compares other CVD practitioners to the skill of that team or teams. Such data could also inform causal reasoning about certain event orderings and help identify effective interventions. For example, might causing $X < F$ be an effective method to improve the chances of $D < A$ in cases where the vendor is slow to produce a fix? Whether it is better to compare the skill of a team to blind luck via the i.i.d. assumption or to other teams via measurement remains an open question.

To address the question, a future research effort must collect and collate a large amount of data about the timing sequences of events in the model for a variety of stakeholder groups and a variety of vulnerabilities. Then deeper analysis using joint probabilities could continue, if the modeling choice is to base skill off of a measure from past observations.

While there is a modeling choice about using the uniformity assumption versus observations from past CVD (see Section 8.1), the model does not depend on whether the uniformity assumption actually holds. We have provided a means to calculate from observations a deviation from the desired “reasonable baseline,” whether this is based on the i.i.d. assumption or not. Although, via our research questions, we have provided a method for evaluating skill in CVD, evaluating the overarching question of *fairness* in MPCVD requires a much broader sense of CVD practices.

MPCVD Criteria Do Not Account for Equitable Resilience. The proposed criteria for MPCVD in Section 8.2 fail to account for either user populations or their relative importance. For example, suppose an MPCVD case had a total of 15 vendors, with 5 vendors representing 95% of the total userbase achieving highly preferred outcomes and 10 vendors with poor outcomes representing the remaining 5% of the userbase. The desired criteria (high median α score with low variance) would likely be unmet even though most users were protected.

Similarly, it is possible that a smaller set of vendor/product pairs represents a disproportionate concentration of the total risk posed by a vulnerability,⁶ and again aggregation across all vendor/product pairs may be misleading. In fact, risk concentration within a particular user population may lead to a need for strategies that appear inequitable at the vendor level while achieving greater outcome equity at a larger scale.

The core issue is that we lack a utility function to map from observed case histories to harm reduction. Potential features of such a function include aggregation across vendors and/or users. Alternatively, it may be possible to devise a method for weighting the achieved histories in an MPCVD case by some proxy for total user risk. Other approaches remain possible as well, such as a heuristic to avoid catastrophic outcomes for all, then apply a weighted sum over the impact to remaining users. Future work might also consider whether criteria other than high median and low variance could be applied. Regardless, achieving accurate estimates of such parameters is likely to remain challenging.

The Model Has No Sense of Timing. There is no concept of time in this model, but delays between events can make a big difference in history results. Two cases in which $F < A$ would be quite different if the time gap between these two events was 1 week versus 3 months, as this gap directly bears on the need for speed in deploying fixes. Organizations may wish to extend this model by setting timing expectations in addition to simple precedence preferences. For example, organizations may wish to specify service level objectives for $V < F$, $F < D$, $F < A$, and so forth.

Furthermore, in the long run the elapsed time for $F < A$ essentially dictates the response time requirements for vulnerability management (VM) processes for system owners. Neither system owners nor vendors get to choose when attacks happen, so we should expect stochasticity to play a significant role in this timing. However, if an organization cannot consistently achieve a shorter lag between F and D than between F and A (i.e., achieving $D < A$) for a sizable fraction of the vulnerability cases they encounter, it is difficult to imagine that organization being satisfied with the effectiveness of their VM program.

Similarly, the model casts each event $e \in E$ as a singular point event, even though some—such as fix deployed D —would be more accurately described as diffusion or multi-agent (as above) processes. To apply this model to real world observations, it may be pragmatic to adapt the event definition to include some defined threshold criteria. A fixed quantile appears to be a reasonable approach. For example, a stakeholder might decide that their goal is for 80% of known systems to be patched. They then could observe the deployed fix ratio for their constituency and mark the event D as having occurred when that threshold is reached.

Attacks as random events. In the model presented here, attacks are modeled as random events, but they are not. At an individual or organization level, attackers are intelligent adversaries and can be expected to follow their own objectives and processes to achieve their ends. However, modeling the details of various attackers is beyond the scope of this model. Thus, we believe that a stochastic approach to adversarial actions is reasonable from the perspective of a vendor or system owner. Furthermore, if attacks were easily predicted, we would be having a very different conversation.

Observation may be limited. Not all events $e \in E$, and therefore not all desiderata $d \in \mathbb{D}$ will be observable by all interested parties. But in many cases at least some are, which can still help to infer reasonable limits on the others, as shown in Section 7.3. Vendors are of course well placed to observe most of the events in each

⁶User concentration is one way to think about risk, but it is not the only way. Value density, as defined in [38] is another.

case, even more so if they have good sources of threat information to bolster their awareness of the X and A events. A vigilant public can also be expected to eventually observe most of the events, although V might not be observable unless vendors, researchers, and/or coordinators are forthcoming with their notification timelines (as many increasingly are). D is probably the hardest event to observe for all parties, for the reasons described in the timing discussion above.

10 RELATED WORK

Numerous models of the vulnerability life cycle and CVD have been proposed. Arbaugh, Fithen, and McHugh provide a descriptive model of the life cycle of vulnerabilities from inception to attacks and remediation [1], which we refined with those of Frei et al. [19], and Bilge and et al. [8] to form the basis of this model as described in Section 2. We also found Lewis' literature review of vulnerability lifecycle models to be useful [27].

Prescriptive models have also been proposed. Christey and Wysopal's 2002 IETF draft laid out a process for responsible disclosure geared towards prescribing roles, responsibilities for researchers, vendors, customers, and the security community [13]. The NIAC Vulnerability Disclosure Framework also laid out a prescriptive process for coordinating the disclosure and remediation of vulnerabilities [12]. The CERT Guide to Coordinated Vulnerability Disclosure provides a practical overview of the CVD process [22]. ISO/IEC 29147 describes standard externally-facing processes for vulnerability disclosure from the perspective of a vendor receiving vulnerability reports, while ISO/IEC 30111 describes internal vulnerability handling processes within a vendor [23, 24]. The **FIRST Product Security Incident Response Team (PSIRT) Services Framework** provides a practical description of the capabilities common to vulnerability response within vendor organizations. The FIRST provides a number of scenarios for MPCVD [18]. Many of these scenarios can be mapped directly to the histories $h \in H$ described here.

Benchmarking CVD capability is the topic of the **Vulnerability Crduoordination Maturity Model (VCMM)** from Luta Security [28]. The Vulnerability Coordination Maturity Model (VCMM) addresses five capability areas: organizational, engineering, communications, analytics, and incentives. Of these, our model is perhaps most relevant to the analytics capability, and could be used to inform an organization's assessment of their progress in this dimension.

Economic analysis of CVD has also been done. Arora et al. explored the CVD process from an economic and social welfare perspective [2–7]. More recently, so did Silfversten et al. [37]. Cavusoglu and Cavusoglu model the mechanisms involved in motivating vendors to produce and release patches [10]. Ellis et al. examine the dynamics of labor market for bug bounties both within and across CVD programs [16]. Lewis highlights systemic themes within the vulnerability discovery and disclosure system [27]. Pupillo et al. explore the policy implications of CVD in Europe [35]. Moore and Householder modeled the interactions of VM and MPCVD processes [29]. A model for prioritizing vulnerability response that considers X and A , among other impact factors, is found in Spring et al. [38].

Other work has examined the timing of events in the lifecycle, sometimes with implications for forecasting. Ozment and Schechter examine the rate of vulnerability reports as software ages [34]. Bilge and Dumitraş study 18 vulnerabilities in which $A < P$, finding a lag of over 300 days [8]. Jacobs et al. propose an Exploit Prediction Scoring System [25], which could provide insight into $V < A$, $F < A$, and other desiderata $d \in \mathbb{D}$. Householder et al. find that while only about 5% of vulnerabilities have public exploits available via commodity tools, for those that do the median lag between P and X was 2 days [20].

Frei et al. describe the timing of many of the events here, including F , D , X , and P and the Δt between them for the period 2000–2007 across a wide swath of industry [19]. In their analysis, they note that $X < P$ in 15% of vulnerabilities they analyzed. That means $f_{P < X}^{obs} = 0.85$, but from Table 4 we already expect $f_{P < X} = 0.5$ from chance alone. So we therefore compute $\alpha_{P < X} = 0.7$. Similarly, they report that a patch is available on or before the date of public awareness in 43% of vulnerabilities (our interpretation of their “zero day patch” where $t_P - t_F = 0$

is that in order for P and F to happen in the same day the patch existed before the P event). In other words, $f_{F < P}^{obs} = 0.43$, giving us an $\alpha_{F < P} = 0.36$ once we factor in $f_{F < P} = 0.111$.

11 CONCLUSION

We have described a model of all possible CVD histories H and defined a set of desired criteria $\mathbb{D}$ that are preferable in each history. This allowed us to create a partially ordered set over all histories and to compute a baseline expected frequency for each desired criteria. We also proposed a new performance indicator for comparing actual CVD experiences against a benchmark, and proposed an initial benchmark based on the expected frequency of each desired criteria. We demonstrated this performance indicator in a few examples, indicating that at least some CVD practices appear to be doing considerably better than random. Finally, we posited a way to apply these metrics to measure the efficacy of MPCVD processes.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers of DTRAP for their helpful and constructive comments.

REFERENCES

- [1] William A. Arbaugh, William L. Fithen, and John McHugh. 2000. Windows of vulnerability: A case study analysis. *Computer* 33, 12 (2000), 52–59.
- [2] Ashish Arora, Jonathan P. Caulkins, and Rahul Telang. 2006. Research note—Sell first, fix later: Impact of patching on software quality. *Management Science* 52, 3 (2006), 465–471.
- [3] Ashish Arora, Chris Forman, Anand Nandkumar, and Rahul Telang. 2010. Competition and patching of security vulnerabilities: An empirical analysis. *Information Economics and Policy* 22, 2 (2010), 164–177.
- [4] Ashish Arora, Ramayya Krishnan, Rahul Telang, and Yubao Yang. 2010. An empirical analysis of software vendors’ patch release behavior: Impact of vulnerability disclosure. *Information Systems Research* 21, 1 (2010), 115–132.
- [5] Ashish Arora, Anand Nandkumar, and Rahul Telang. 2006. Does information security attack frequency increase with vulnerability disclosure? An empirical analysis. *Information Systems Frontiers* 8, 5 (2006), 350–362.
- [6] Ashish Arora and Rahul Telang. 2005. Economics of software vulnerability disclosure. *IEEE Security & Privacy* 3, 1 (2005), 20–25.
- [7] Ashish Arora, Rahul Telang, and Hao Xu. 2008. Optimal policy for software vulnerability disclosure. *Management Science* 54, 4 (2008), 642–656.
- [8] Leyla Bilge and Tudor Dumitras. 2012. Before we knew it: An empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. 833–844.
- [9] Lawrence D. Brown, T. Tony Cai, and Anirban DasGupta. 2001. Interval estimation for a binomial proportion. *Statistical Science* 16, 2 (2001), 101–133.
- [10] Hasan Cavusoglu, Huseyin Cavusoglu, and Srinivasan Raghunathan. 2007. Efficiency of vulnerability disclosure mechanisms to disseminate vulnerability knowledge. *IEEE Transactions on Software Engineering* 33, 3 (2007), 171–185.
- [11] CERT Coordination Center (CERT/CC). 2020. CERT Vulnerability Data Archive. Retrieved from <https://github.com/CERTCC/Vulnerability-Data-Archive>. (2020). Accessed: 2020-06-08.
- [12] John T. Chambers and John W. Thomson. 2004. National Infrastructure Advisory Council’s Vulnerability Disclosure Framework: Final Report and Recommendations. Retrieved from <https://www.cisa.gov/publication/niac-vulnerability-framework-final-report>. (2004). Accessed: 2020-07-27.
- [13] Steve Christey and Chris Wysopal. 2002. Responsible Vulnerability Disclosure Process. Retrieved from <https://tools.ietf.org/html/draft-christey-wysopal-vuln-disclosure-00>. (February 2002). <https://tools.ietf.org/html/draft-christey-wysopal-vuln-disclosure-00>. Accessed: 2020-07-27.
- [14] Cyentia Institute. 2019. *Prioritization to Prediction Volume 2: Getting Real About Remediation*. Technical Report. Cyentia Institute, LLC.
- [15] Marcel Dreef, Peter Borm, and Ben Van der Genugten. 2004. Measuring skill in games: Several approaches discussed. *Mathematical Methods of Operations Research* 59, 3 (2004), 375–391.
- [16] Ryan Ellis, Keman Huang, Michael Siegel, Katie Mousouris, and James Houghton. 2017. *New Solutions for Cybersecurity*. MIT Press, Chapter Fixing a hole: The labor market for bugs., 122–147.
- [17] Benjamin Eva. 2019. Principles of Indifference. *The Journal of Philosophy* 116, 7 (2019), 390–411.
- [18] Forum of Incident Response and Security Teams. 2020. Guidelines and Practices for Multi-Party Vulnerability Coordination and Disclosure. Retrieved from <https://www.first.org/global/signs/vulnerability-coordination/multi-party-guidelines-v1.1>. (2020). Accessed: 2020-07-27.

- [19] Stefan Frei, Dominik Schatzmann, Bernhard Plattner, and Brian Trammell. 2010. Modeling the security ecosystem-the dynamics of (in) security. In *Proceedings of the Economics of Information Security and Privacy*. Springer, 79–106.
- [20] Allen D. Householder, Jeff Chrabaszcz, Trent Novelty, David Warren, and Jonathan M. Spring. 2020. Historical analysis of exploit availability timelines. In *Proceedings of the 13th USENIX Workshop on Cyber Security Experimentation and Test*.
- [21] Allen D. Householder and Jonathan Spring. 2021. *A State-Based Model for Multi-Party Coordinated Vulnerability Disclosure (MPCVD)*. Technical Report CMU/SEI-2021-SR-021. Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, PA.
- [22] Allen D. Householder, Garret Wassermann, Art Manion, and Chris King. 2017. *The CERT Guide to Coordinated Vulnerability Disclosure*. Technical Report CMU/SEI-2017-SR-022. Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, PA.
- [23] ISO. 2018. *Information technology — Security techniques — Vulnerability disclosure*. Standard 29147:2018. International Organization for Standardization, Geneva, CH.
- [24] ISO. 2019. *Information technology — Security techniques — Vulnerability handling processes*. Standard 30111:2019. International Organization for Standardization, Geneva, CH.
- [25] Jay Jacobs, Sasha Romanosky, Benjamin Edwards, Michael Roytman, and Idris Adjerid. 2021. Exploit prediction scoring system (EPSS). *Digital Threats* 2, 3 (2021).
- [26] Patrick Larkey, Joseph B. Kadane, Robert Austin, and Shmuel Zamir. 1997. Skill in games. *Management Science* 43, 5 (1997), 596–609.
- [27] Paul Simon Lewis. 2017. The global vulnerability discovery and disclosure system: A thematic system dynamics approach. PhD Dissertation, Cranfield University.
- [28] Luta Security. 2020. Vulnerability Coordination Security Model. Retrieved from <https://www.lutasecurity.com/vcmm>. (2020). Accessed: 2020-09-17.
- [29] Andrew P. Moore and Allen D. Householder. 2019. *Multi-Method Modeling and Analysis of the Cybersecurity Vulnerability Management Ecosystem*. Technical Report. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.
- [30] National Cyber Security Centre. 2018. *Coordinated Vulnerability Disclosure: The Guideline*. Technical Report. National Cyber Security Centre, Netherlands.
- [31] Lily Hay Newman. 2018. Senators Fear Meltdown and Spectre Disclosure Gave China an Edge. Retrieved from <https://www.wired.com/story/meltdown-and-spectre-intel-china-disclosure/>, Jul 2018. Accessed: 2022-06-07.
- [32] NIST. 2020. National Vulnerability Database. Retrieved from <https://nvd.nist.gov>. (2020). Accessed: 2020-06-08.
- [33] Offensive Security. 2020. Exploit DB. Retrieved from <https://github.com/offensive-security/exploitdb>. (2020). Accessed: 2020-06-08.
- [34] Andy Ozment and Stuart E. Schechter. 2006. Milk or wine: Does software security improve with age?. In *Proceedings of the USENIX Security Symposium*.
- [35] Lorenzo Pupillo, Afonso Ferreira, and Gianluca Varisco. 2018. Software Vulnerability Disclosure in Europe: Technology, policies and legal challenges, report of a CEPS Task Force. Report ISBN 978-94-6138-687-8, Center for European Policy Studies (CEPS), Place du Congrès 1, B-1000 Brussels, June 2018.
- [36] Rapid7. 2020. Metasploit Framework. Retrieved from <https://github.com/rapid7/metasploit-framework>. (2020). Accessed: 2020-06-08.
- [37] Erik Silfversten, William D. Phillips, Giacomo Persi Paoli, and Cosmin Ciobanu. 2018. Economics of vulnerability disclosure. Report TP-07-18-080-EN-N, European Union Agency for Network and Information Security (ENISA), 1 Vasilissis Sofias, Marousi 151 24, Attiki, Greece, December 2018. DOI: [10.2824/49807](https://doi.org/10.2824/49807)
- [38] Jonathan M. Spring, Eric Hatleback, Allen D. Householder, Art Manion, and Deana Shick. 2020. Prioritizing vulnerability response: A stakeholder-specific vulnerability categorization. In *Proceedings of the Workshop on the Economics of Information Security*. Brussels, Belgium.
- [39] Jeroen van der Ham and Shawn Richardson. 2020. Ethics for Incident Response and Security Teams. Retrieved from <https://www.first.org/global/sigs/ethics/ethics-first>. (Jun. 2020). Accessed: 2020-03-10.
- [40] Yang Xiao, Bihuan Chen, Chendong Yu, Zhengzi Xu, Zimu Yuan, Feng Li, Binghong Liu, Yang Liu, Wei Huo, Wei Zou, and Wenchang Shi. 2020. MVP: Detecting vulnerabilities using patch-enhanced vulnerability signatures. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security'20)*. 1165–1182.
- [41] Yifei Xu, Zhengzi Xu, Bihuan Chen, Fu Song, Yang Liu, and Ting Liu. 2020. Patch based vulnerability matching for binary programs. In *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 376–387.

Received 20 November 2020; revised 8 April 2021; accepted 22 July 2021