Historical Analysis of Exploit Availability Timelines

(research paper, long)
CSET
August 10, 2020

Allen D. Householder, Jeff Chrabaszcz,
Trent Novelly, David Warren,
Jonathan M. Spring

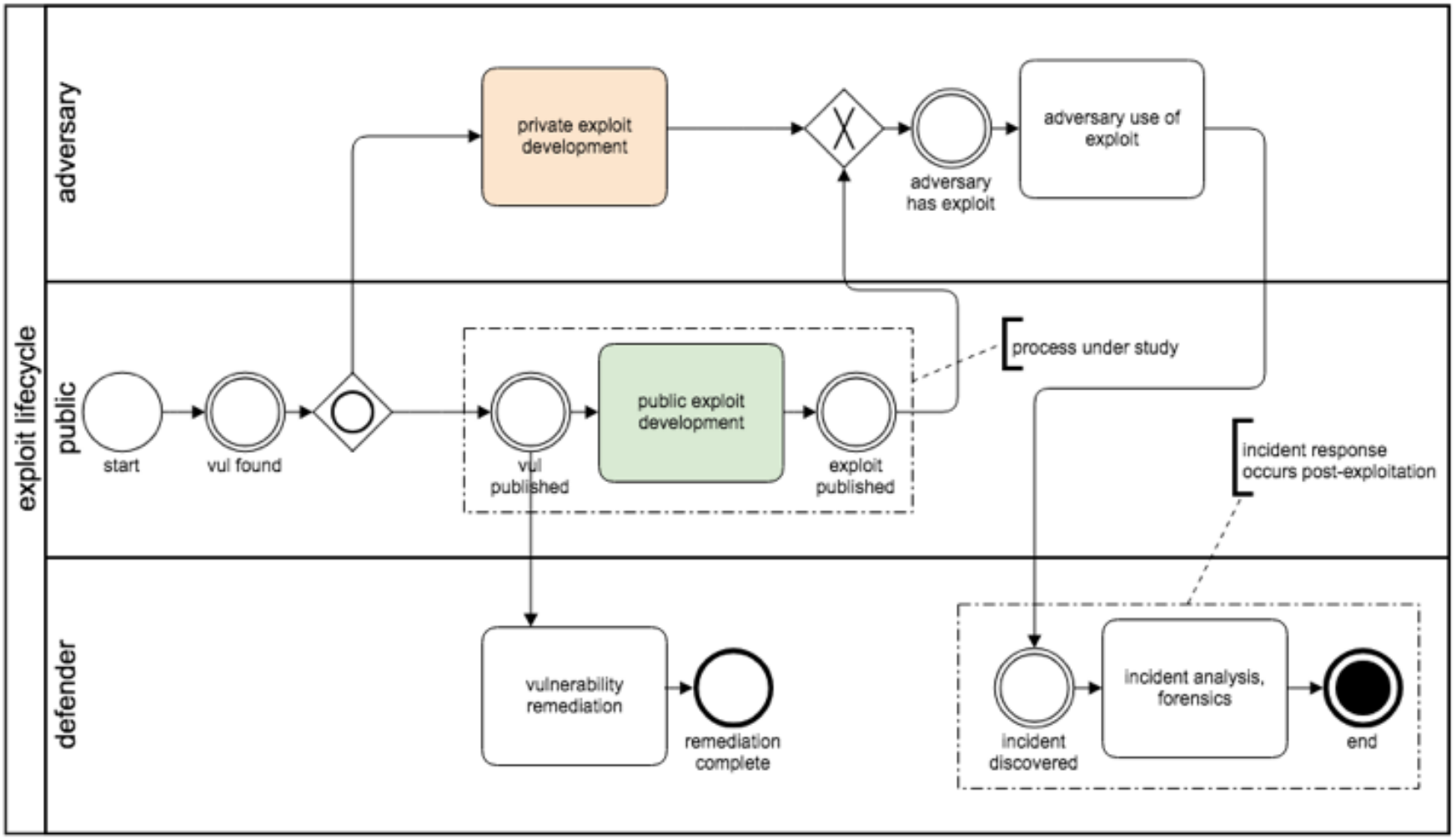
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Document Markings

This material is based upon work funded and supported by the Department of Defense 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.

DM20-0572

Exploit Development Lifecycle



Research questions

1. How many CVE-IDs (Common Vulnerabilities and Exposures) get public exploit code within 365 days of publication?
2. Of those CVE-IDs that get public exploit code, how quickly is such code published? Do different sub-populations of vulnerabilities, as defined by the Common Weakness Enumeration (CWE), have different publication rates?
3. Are there any features of a CVE-ID that are correlated with likelihood of public exploit code or speed of its publication?

Background

- We want to use exploit publication as part of prioritization and triage of work items related to vulnerabilities[1]
- There is a long history of both practitioners[2,3] and researchers[4,5] using either active exploitation or public exploit code as part of vulnerability priority
- But none of this work has presented a data-driven, historical view of what has influenced exploit code publication during a long time window

[1] Jonathan M Spring, Eric Hatleback, Allen D. Householder, Art Manion, and Deana Shick. Prioritizing vulnerability response: A stakeholder-specific vulnerability categorization. In *Workshop on the Economics of Information Security*, Brussels, Belgium, December 2020.

[2] Dan Guido. A case study of intelligence-driven defense. *IEEE security & privacy*, 9(6):67–70, 2011.

[3] Jay Jacobs, Sasha Romanosky, Benjamin Edwards, Michael Roytman, and Idris Adjerid. Exploit prediction scoring system (EPSS). In *Workshop on the Economics of Information Security*, Boston, MA, June 2019.

[4] Luca Allodi and Fabio Massacci. Comparing vulnerability severity and exploits using case-control studies. *Transactions on Information and System Security*, 17(1):1–20, 2014.

[5] Carl Sabottke, Octavian Suciu, and Tudor Dumitras. Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In *USENIX Security*, pages 1041–1056, Washington, D.C., August 2015.

Methodology – Survival analysis

We use survival analysis,[1] by metaphor to hazard analysis in medicine, to estimate the impact of different factors on the "survival" of a CVE-ID

- A CVE-ID "survives" as long as no public exploit code is associated with it
- Publication date is ≥ 0, because we consider publication of exploit code tagged with the CVE-ID as de facto public disclosure

The statistical model provides an interval estimate of how different features effect the survival of CVE-IDs in the population

- Model assumes a proportional (multiplicative) relationship for the effect of each variable

[1] David R Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

Data sources

CVE-ID publication date and other data (2013-2020)

- National Vulnerability Database
- MITRE (for CVE, CWE, and CPE)
- CERT/CC vulnerability notes
- The Zero Day Initiative

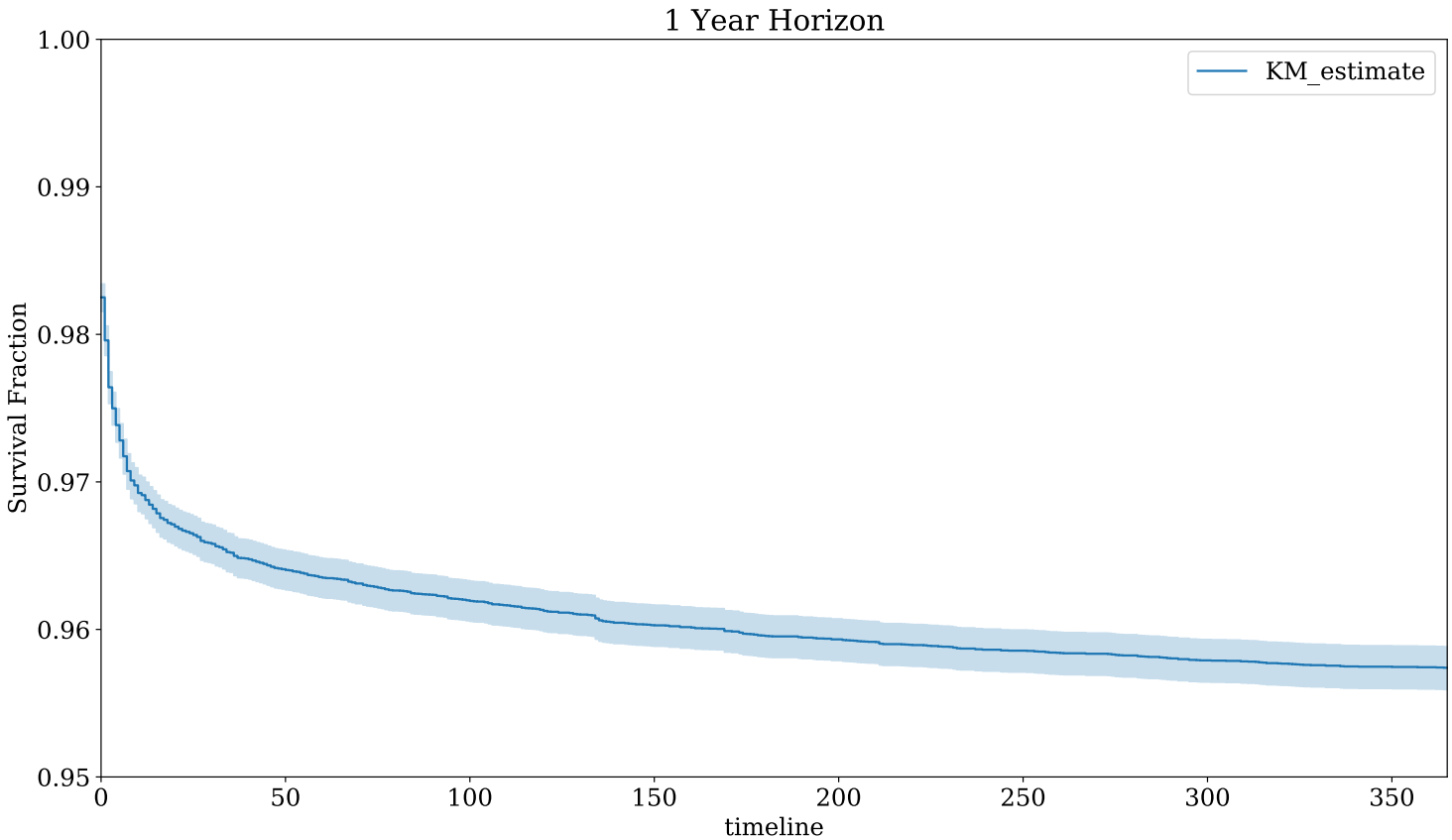
Exploit code publication date

- GitHub repositories for:
  - Exploit Database
  - Metasploit Framework

Data challenges:

- Identifying equivalent exploits
- Quality of manual tags

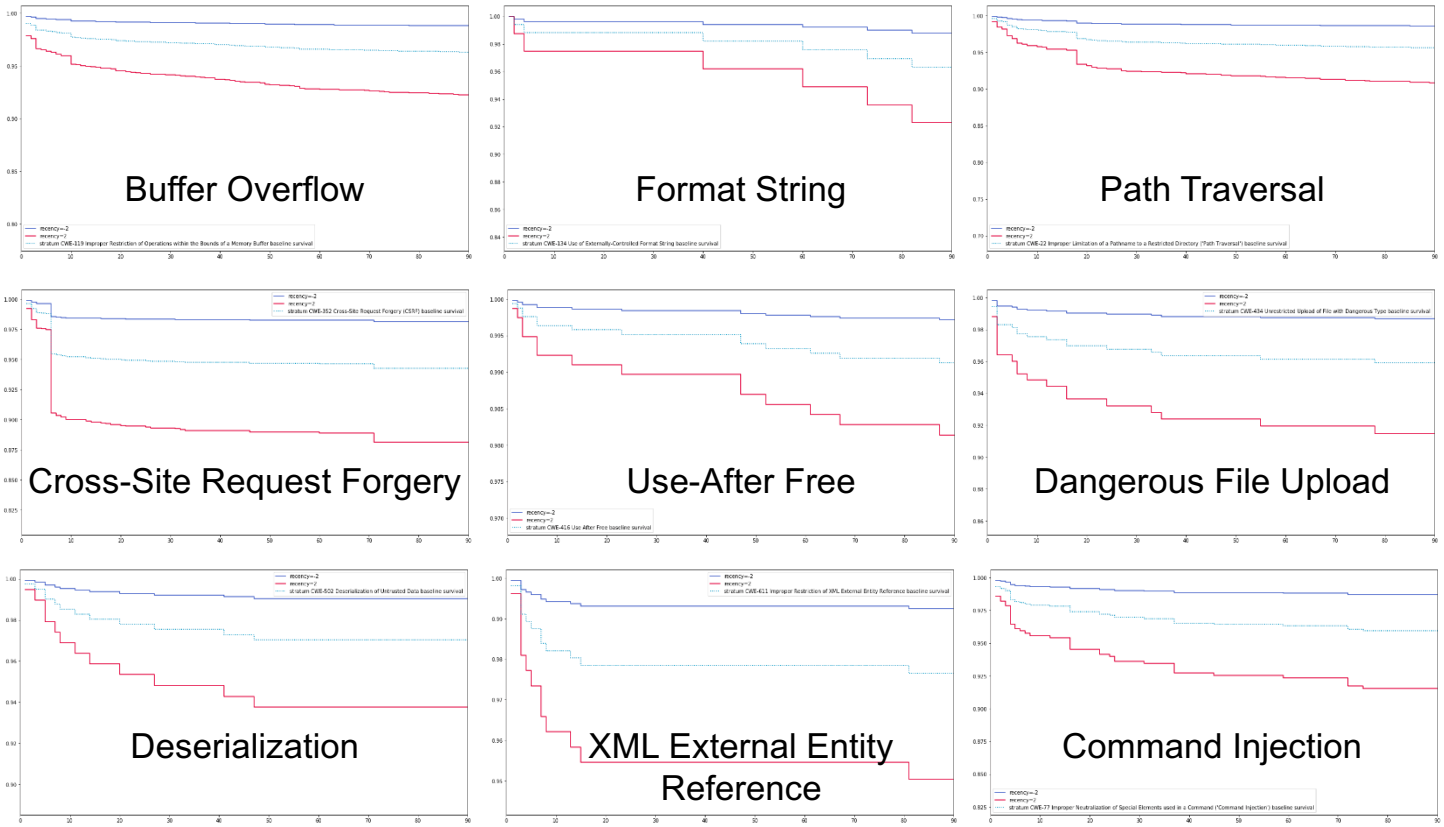
Results (all CVE-IDs)



Results (RQ1 and RQ2)

1. How many CVE-IDs?
   - 4.1%, +/- 0.1%
     - ~3,100 of ~75,800 CVE-IDs studied
   - ~42% of exploits published occur on the same day as disclosure
2. How quickly is such code published?
   - Median: 2 days;
   - Mean: 91.5 days.
   - Different CWEs have different publication rates. E.g.:
     - CWE-400 (uncontrolled resource consumption) is 2-9 times less likely
     - CWE-113 (improper CRLF neutralization in HTTP headers) is 60-90 times more likely

Exploit Availability Accelerates with CWE Age



Blue/Top Line: Older
Red/Bottom Line: Newer
Teal/Middle Line: Baseline

Results (RQ3)

3. Correlated features of a CVE-ID?
   - CVSSv3 base score – higher score is 1.2 to 1.3 times more likely to see a exploit publication
   - Number of CVE-IDs in the CWE category – larger category is 1.15 to 1.4 times more likely
   - CVE-ID publication date – more recently published CVE-IDs are 1.23 to 1.32 times more likely
   - Vendor – some vendors are more likely to have exploits published for their products than others; mostly, these are vendors of security products
   - Number of vendors involved in vulnerability – no clear correlation between multi-party cases and exploitation

Interpretation

Good: Relatively few CVE-IDs have exploit code published

Good: Exploit search could be an automated part of prioritization

Bad: CVE-IDs with exploit code often have it published quickly

Bad: Exploit code publication speed is increasing over time

One consistent interpretation is that the features that increase hazard (exploit code publication likelihood) are proxies for something like the value to attackers of the vulnerable component

- We lack a direct measure of value to attackers to check this

Limitations

Undercounting

- Not all vulnerabilities get CVE-IDs
- Not all exploit code is published
- Not all public exploit code is collated in ExploitDB and Metasploit
- Not all ExploitDB/Metasploit modules contain a CVE-ID

Practical application

- Not all ExploitDB/Metasploit modules are used by attackers
- We are only able to measure correlation, not causation

Going forwards

Try to assess latent variable of “value of the vulnerable component to the adversary”

Integrate automated exploit code search into vulnerability triage and prioritization

Assess categories of vulnerabilities that may not receive a CVE-ID or where exploit code is a configuration of some open-source tool

- Attacks related to TLS certificates by an on-path attacker, for example, can be done by configuring a HTTP proxy

Thanks for your time!

Questions?

Contact jspring AT sei cmu edu

Code:

https://github.com/CERTCC/git_vul_driller