8.1 Vulnerability IDs and DBs

The units of work in CVD are vulnerability reports or cases. However, a single case may actually address multiple vulnerabilities. Teasing out how many problems are involved in a report can be tricky at times. The implications of this in terms of the CVD process and the compilation of vulnerability databases is significant.

This section is adapted from a CERT/CC blog post by Householder [1].

On the Complexities of Vulnerability Identity

Vulnerability identifiers can serve multiple purposes.

They may be used to identify the following:

- A vulnerability report or case
- A document or database entry that describes a vulnerability (e.g., CERT Vulnerability Notes [2])
- The specific flaw that such a document or report describes [3]

Now this isn't really a problem as long as one case describes one vulnerability and that case results in the creation of one document. But that's not always the case, for a number of reasons, including those below:

- Different processes use different abstractions to define what "unit vulnerability" is. For example, CVE has specific guidance on counting rules [4].
- It's rare for vendors to release single-issue patches. More often they prefer to roll up multiple fixes into a single release, and then publish a document about the release [5].
- In the case of independent discovery, or at least duplicate reporting, multiple cases may be opened describing the same vulnerability. In some instances, this fact may not become obvious until considerable effort has been put into isolating the bugs in each report. For example, a single vulnerability can manifest in different ways depending on how it's triggered. The connection might only be discovered on root cause analysis.
- Automated testing such as fuzzing can lead to rapid discovery of very large numbers of unique failure cases that are difficult to resolve into specific bugs.

Automated testing can also identify so many individual vulnerabilities that human-oriented case handling processes cannot scale to treat each one individually. Here's an extreme example of this phenomenon: although the CERT/CC published only a single Vulnerability Note for Android apps that failed to validate SSL certificates, in the end it covered 23,667 vulnerable apps [6,7]. Should each get its own identifier? Yes, and we did assign individual VU# identifiers to each vulnerable app. But this highlights the distinction between the vulnerability and the document that describes it.

As of this writing, work is underway within the Vulnerability Report Data Exchange special interest group (VRDX-SIG) within FIRST [8] on a vulnerability report cross-reference data model that will allow for the expression of relationships between vulnerability reports. The current work in progress can be found at, https://github.com/FIRSTdotorg/vrdx-sig-vxref-wip.

In order to make it easier to relate vulnerability reports and records to each other, the VRDX work represents the following concepts: "possibly related," "related," "not equal," "equal," "superset," "subset," and "overlap."

What CVE Isn't

Because of the prevalence and popular use of CVE IDs in the vulnerability response space, many people assume that vulnerability identity is synonymous with Common Vulnerabilities and Exposures (CVE) [9]. However, let's briefly look at some ways in which that assumption is inaccurate:

- CVE has limited scope of coverage.
- Not all known vulnerabilities are assigned a CVE ID.
- Not all vulnerabilities assigned a CVE ID have a corresponding record in the CVE database.

Every Vulnerability Database Makes Choices

As the CERT/CC's vulnerability analysis efforts have expanded into vulnerability coordination for non-traditional computing products (mobile, vehicles, medical devices, IoT, etc.) [10], we've also begun to hit up against another set of issues affecting vulnerability identities and compatibility across vulnerability databases (VDBs): namely, bias.

Steve Christey Coley and Brian Martin mention a number of biases that affect all VDBs in their BlackHat 2013 talk [11]:

- **Selection bias**. Not all products receive equal scrutiny. Not all vul reports are included in VDBs.
- **Publication bias**. Not all results get published. Some vuls are found but never reported to anyone.
- **Abstraction bias**. This bias is an artifact of the process that VDBs use to assign identifiers to vulnerabilities. (Is it 1 vul or 3?, 23,667 or 1?)
- **Measurement bias**. This bias encompasses errors in how a vulnerability is analyzed, verified, and catalogued.

In an ideal scientific world, bias can be factored into analytical results based on the data collected. But VDBs don't exist solely in the service of scientific purity. Every vulnerability database or catalog makes choices driven by the business requirements and organizational environments in which those VDBs operate.

These choices include the following:

1. **Sources of vulnerability information monitored.** Monitoring all the potential sources of vulnerability information is unrealistic for resource-constrained VDBs; to date we have found none that are not so constrained. This choice is one source of selection bias.
2. **Inclusion and exclusion criteria.** Rules that define what subset of records from the sources monitored will be included (or not) in the VDB must be decided. What kind of vulnerabilities does the VDB track? Is it platform specific? Is it just a single vendor collecting reports in its own products? Is it focused on a particular business sector? This choice is another source of selection bias.
3. **Content detail.** How much (and what kind of) detail goes into each record in a VDB is something that must be decided: for example, whether to include exploit information, workarounds, detection criteria, and so forth.
4. **Abstraction.** What is a "unit" vulnerability? Does this report represent one vul or many? That choice depends on what purpose the VDB serves. Christey and Martin cover this issue in their list of biases, describing it as "the most prevalent source of problems for analysis." CVE has made their abstraction content decision guidance available [12].
5. **Uncertainty tolerance.** How certain is the information included in the record? Is the goal of the VDB to be authoritative on first publication? Or can it tolerate being wrong sometimes in favor of getting things out more quickly?
6. **Latency tolerance.** How quickly do new records need to be placed in the VDB following the initial disclosure? This choice is a distinct tradeoff with uncertainty: consider the differences between breaking news coverage and a history book.
7. **Capacity constraints.** For a VDB, incoming vulnerability report volume is unconstrained while the capacity to consume and process those reports into database records is decidedly not (especially with humans in the loop, and as of this writing they still are).
8. **Users and consumers of the data.** Ultimately, a VDB must serve some useful purpose to some audience in order for it to continue to exist. There is a wide variety of uses for the information contained in VDBs (vulnerability scanning, vulnerability management systems, long-term trend analysis, academic research, quality improvement efforts, supporting acquisition or purchasing decisions, evaluating vendor process effectiveness, etc.), so it shouldn't be surprising that user requirements can drive many of the other choices the VDB operators have to make.

It's important to note that even if two vulnerability databases agree on the first four items in the list above (sources to watch, inclusion criteria, content detail, and abstraction), over time it's easy to wind up with completely distinct data sets due to the latter items (uncertainty tolerance, latency tolerance, capacity constraints, and user needs).

Where We Are vs. Where We Need to Be

The vulnerability databases you are probably most familiar with, such as the National Vulnerability Database (NVD) [13], Common Vulnerabilities and Exposures (CVE) [14], and the CERT Vulnerability Notes Database [15] have historically focused on vulnerabilities affecting traditional computing platforms (Windows, Linux, OS X, and other Unix-derived operating systems) with only a smattering of coverage for vulnerabilities in other platforms like mobile or embedded systems, websites, and cloud services. In the case of websites and cloud services this gap may be acceptable since most such services are effectively single instances of a large-scale distributed system and therefore only the service provider needs to apply a fix. In those cases, there might not be a need for a common identifier since nobody is trying to coordinate efforts across responsible parties. But in the mobile and embedded spaces, we definitely see the need for identifiers to serve the needs of both disclosure coordination and patch deployment.

Furthermore, there is a strong English language and English-speaking country bias in the major U.S.-based VDBs (hopefully this isn't terribly surprising). China has not one but two major VDBs: China National Vulnerability Database of Information Security (CNNVD) [16] and China National Vulnerability Database (CNVD) [17]. We have been working with CSIRTs around the world (e.g., JPCERT/CC [18] and NCSC-FI [19]) to coordinate vulnerability response for years and realize the importance of international cooperation and interoperability in vulnerability response. Given all the above, and in the context of the surging prevalence of bug bounty programs, it seems likely that in the coming years there will be more, not fewer VDBs around the world than there are today. We anticipate those VDBs will cover more products, sectors, languages, countries, and platforms than VDBs have in the past.

Coordinating vulnerability response at local, national, and global scales requires that we have the means to relate vulnerability reports to each other, regardless of the process that originated them. Furthermore, whether they are driven by national, commercial, or sector-specific interests, there will be a need for interoperability across all those coordination processes and the VDBs into which they feed.

Vulnerability IDs, Fast and Slow

Over time, it has become clear that the days of the "One Vulnerability ID to Rule Them All" are coming to a close and we need to start planning for that change. As we've covered above, one of the key observations we've made has been the growing need for multiple vulnerability identifiers and databases that serve different audiences, support diverse business practices, and operate at different characteristic rates.

In his book Thinking, Fast and Slow, Daniel Kahneman describes human thought processes in terms of two distinct systems [20]:

- System 1: Fast, automatic, frequent, emotional, stereotypic, subconscious
- System 2: Slow, effortful, infrequent, logical, calculating, conscious

Making the analogy to CVD processes, notice that historically there has been a need for slower, consistently high-quality, authoritative vulnerability records, trading off higher latency for lower noise. Deconfliction of duplicate records happens before an ID record (e.g., a CVE record) is issued, and reconciliation of errors can be difficult. To date, this practice is the ideal for which many VDBs have strived. Those VDBs remain a valuable resource in the defense of systems and networks around the globe.

Yet there is a different ideal, just as valid: one in which vulnerability IDs are assigned quickly, possibly non-authoritatively, and based on reports of variable quality. This process looks more like "issue, then deconflict." For this new process to work well, post-hoc reconciliation needs to become easier. If you're familiar with the gitflow process [21] in software development, you might recognize this distinction as analogous to the one between the _develop_ and _master_ branches of a software project. The bulk of the work happens in and around the _develop_ branch, and only when things have settled out does the _master_ branch get updated (and merge conflicts are as inevitable as death and taxes).

A Path Toward VDB Interoperability

As mentioned above, the FIRST VRDX-SIG is working on a vulnerability cross-reference scheme that would allow for widely distributed vulnerability ID assignments and VDBs to run at whatever rate is necessary, while enabling the ability to reconcile them later once the dust clears:

- When necessary, the CVD process could operate in System 1 for quick response, and clean up any resulting confusion afterwards.
- A tactical response-focused VDB might be able to tolerate more uncertainty in trade for lower latency.
- A VDB with more academic leanings could do a deep-dive analysis on root causes in exchange for having fewer records and higher latency.

The main idea was that VDB records can be related to each other in one of the following ways:

- equality and inequality (two records describe the same vulnerability or vulnerabilities, or they refer to different ones)
- superset and subset (one record is more abstract than the other)
- overlap (related but not fully contained)

This work builds on both prior work at the CERT/CC and Harold Booth and Karen Scarfone's October 2013 IETF Draft Vulnerability Data Model [22]. However, while it would be great if we could get to a unified data model like the IETF draft for vulnerability information exchange eventually, for now the simplest thing that could possibly work seemed to be coming up with a way to relate records within or between vulnerability databases that explicitly addresses the choices and biases described above. The unified data model might be a longer way off, and we were anticipating the need to reconcile VDBs much sooner.

Looking Ahead

Everything we have discussed in this section is work in progress, and some things are changing rapidly on a number of related fronts. Nevertheless, while it's hard to say how we'll get there, it seems inevitable that we'll eventually reach a point where vulnerability IDs can be issued (and deconflicted) at the speed necessary to improve coordinated global vulnerability response while maintaining our ability to have high-quality, trusted sources of vulnerability information.

Here in the CERT/CC Vulnerability Analysis team, we recognize the need for slower, "correct-then-issue" vulnerability IDs as well as faster moving "issue-then-correct" IDs. We believe that there is room for both (and in fact many points in between). Our goal in participating in the VRDX-SIG is to enable interoperability between any willing VDBs. We intend to continue our efforts to build a better way forward that suits everyone who shares our interest in seeing that vulnerabilities get coordinated and disclosed, and that patches are created and deployed, all with an eye toward minimizing societal harm in the process.

< 8. Open Problems in CVD | 8.2 IoT and CVD >

References

1. A. Householder, "Vulnerability IDs, Fast and Slow," 11 March 2016. [Online]. Available: https://insights.sei.cmu.edu/cert/2016/03/vulnerability-ids-fast-and-slow.html. [Accessed 7 June 2017].
2. CERT/CC, "Vulnerability Notes Database," [Online]. Available: https://www.kb.cert.org/vuls. [Accessed 16 May 2017].
3. MITRE, "Common Vulnerabilities and Exposures," [Online]. Available: https://cve.mitre.org/. [Accessed 16 May 2017].
4. MITRE, "Common Vulnerabilities and Exposures (CVE) Numbering Authority (CNA) Rules Version 1.1," 16 September 2016. [Online]. Available: https://cve.mitre.org/cve/cna/CNA_Rules_v1.1.pdf. [Accessed 16 May 2017].
5. N. Mercer, "Further simplifying servicing models for Windows 7 and Windows 8.1," 15 August 2016. [Online]. Available: https://blogs.technet.microsoft.com/windowsitpro/2016/08/15/further-simplifying-servicing-model-for-windows-7-and-windows-8-1/. [Accessed 24 May 2017].
6. W. Dormann, "Vulnerability Note VU#582497 Multiple Android applications fail to properly validate SSL certificates," CERT/CC, 3 September 2014. [Online]. Available: https://www.kb.cert.org/vuls/id/582497. [Accessed 16 May 2017].
7. W. Dormann, "Android apps that fail to validate SSL," 29 August 2014. [Online]. Available: https://docs.google.com/spreadsheets/d/1t5GXwjw82SyunALVJb2w0zi3FoLRIkfGPc7AMjRF0r4. [Accessed 16 May 2017].
8. FIRST, "Vulnerability Reporting and Data eXchange SIG (VRDX-SIG)," [Online]. Available: https://www.first.org/global/sigs/vrdx. [Accessed 16 May 2017].
9. MITRE, "Common Vulnerabilities and Exposures," [Online]. Available: https://cve.mitre.org/. [Accessed 16 May 2017].
10. D. Klinedinst, "Coordinating Vulnerabilities in IoT Devices," 27 January 2016. [Online]. Available: https://insights.sei.cmu.edu/cert/2016/01/coordinating-vulnerabilities-in-iot-devices.html. [Accessed 16 May 2017].
11. S. Christey Coley and B. Martin, "Buying Into the Bias: Why Vulnerability Statistics Suck," in *BlackHat*, 2013.
12. MITRE, "CVE Abstraction Content Decisions: Rationale and Application," 15 June 2005. [Online]. Available: https://cve.mitre.org/cve/editorial_policies/cd_abstraction.html. [Accessed 24 May 2017].
13. National Institute of Standards and Technology, "National Vulnerability Database," [Online]. Available: https://nvd.nist.gov/. [Accessed 16 May 2017].
14. MITRE, "Common Vulnerabilities and Exposures," [Online]. Available: https://cve.mitre.org/. [Accessed 16 May 2017].
15. CERT/CC, "Vulnerability Notes Database," [Online]. Available: https://www.kb.cert.org/vuls. [Accessed 16 May 2017].
16. CNNVD, "China National Vulnerability Database of Information Security," [Online]. Available: http://www.cnnvd.org.cn/. [Accessed 16 May 2017].
17. CNVD, "China National Vulnerability Database," [Online]. Available: http://www.cnvd.org.cn/. [Accessed 16 May 2017].
18. JPCERT/CC, "Japan Computer Emergency Response Team Coordination Center," [Online]. Available: https://www.jpcert.or.jp/english/. [Accessed 16 May 2017].
19. NCSC-FI, "Finnish Communications Regulatory Authority / National Cyber Security Centre Finland," [Online]. Available: https://www.viestintavirasto.fi/en/cybersecurity.html.

20. D. Kahneman, Thinking, Fast and Slow, Macmillan, 2011.
21. V. Driessen, "A successful Git branching model," 5 January 2010. [Online]. Available: http://nvie.com/posts/a-successful-git-branching-model/. [Accessed 16 May 2017].
22. H. Booth and K. Scarfone, "Vulnerability Data Model draft-booth-sacm-vuln-model-02," 25 April 2013. [Online]. Available: https://tools.ietf.org/html/draft-booth-sacm-vuln-model-02. [Accessed 16 May 2107].