# For Good Measure

## To Burn or Not to Burn

DAN GEER AND JON CALLAS

Dan Geer is the CISO for In-Q-Tel and a security researcher with a quantitative bent. He has a long history with the USENIX Association, including officer positions, program committees, etc. dan@geer.org

Jon Callas is a cryptographer, software engineer, UX designer, and entrepreneur. He is the co-author of many crypto and security systems, including OpenPGP, DKIM, ZRTP, Skein, and Threefish. He has co-founded several startups, including PGP, Silent Circle, and Blackphone. He has worked on security, UX, and crypto for Tesla, Kroll-O'Gara, Counterpane, and Entrust. He is fond of Leica cameras, Morgan sports cars, and Birman cats. His photographs have been used by Wired, CBS News, and The Guggenheim Museum. jon@callas.org

There is no question that vulnerabilities are important. There is a rich history of vulnerabilities and of their use, yet if that history is a signal, then it is a noisy one. Inferences drawn from agreed upon history of vulnerabilities are still the source of quite conflicting interpretations—proof that it is hard to reduce the question of vulnerabilities to a simple set of inferences. Experts are less likely to agree on a simple set of inferences about vulnerabilities than the non-experts. Often as not, experts claim that all other expert opinions besides theirs are simplistic rather than simple (and that "I have discovered a truly remarkable proof which this [Tweet] is too small to contain").

At the time of writing this column, a new report from RAND had just appeared. The RAND report [1] (which you *must* read) is the best look yet at the question of vulnerabilities as seen through the lens of vulnerabilities not yet known. As should be expected, a part of its conclusions were immediately dismissed as simplistic by Those Who Tweet.

Three terms from the field of epidemiology may help us think about the life cycle of vulnerabilities. First is *incidence* ($I$), which is the number of new cases of disease which appear per unit time. Second is *prevalence* ($P$), which is the number of infections at a given time. Third is *duration* ($D$), which is the time interval between when infection appears and when that infection is cured. In a stable population, those three are mutually redundant—knowing any two of them allows you to determine the third: $I*D=P$. For the defender whose job is to treat disease, prevalence is the measure of workload. For the defender whose job is to prevent disease, changes in incidence are the measure of whether their work has or has not been successful. For the defender whose job is to judge the societal cost of disease, duration is likely the focus. Analysts studying risk factors for the disease must use incident cases within a given time interval rather than prevalent cases at a given time, i.e., near real-time detection matters: cf., "Neyman" bias.

But each of $I$, $P$, and $D$ are for the single disease case. Thinking of the offender as an opportunistic infection, that is to say a secondary (intentional) infection that exploits an already infected patient, a patient whose existing infection(s) make that patient susceptible (vulnerable) to additional infections, the defender might come to think in terms of global immune system failure more than the lack of some specific antibody. Just as there is no human immune to all human diseases, RAND notes that

> [a]ny serious attacker can always get an affordable zero-day for almost any target. The majority of the cost of a zero-day exploit does not come from labor, but rather the value inherent in them and the lack of supply.

Which reminds us that we are talking about sentient opponents, not stray alpha particles or metal fatigue. We are inside a natural experiment, not some controlled laboratory work.

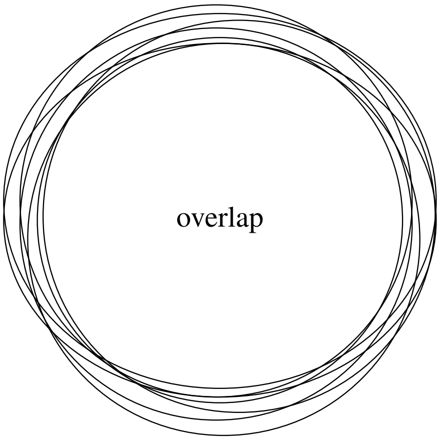

**Figure 1:** The threat landscape includes the overlap of different actors' toolsets.

So be it, but it is that existence of multiple possible infections—due to multiple possible infectious agents—that we find worth comment. Consider the set of tools (vulnerabilities, exploits, software, etc.) that *A* has, and the set that *B* has, and *C*, and *D*, and *E*,...Make a Venn diagram of these; their union is the threat landscape (Figure 1). What is especially noteworthy is not the area common to all the closed curves in the diagram but the areas outside that joint intersection.

If any actor holds their tools secret, then what is in that actor's subset that is not in the union of everyone else? If that subset is small compared to the whole set union, and then the marginal cost to you of any actor, state or otherwise, holding such a cache of tools is small. Yet to any given attacker, they view their whole tool set as an asset and are loathe to give it up. Thus, perhaps counterintuitively, they tend to view their tool set as being worth a lot because it is their whole set, but the defender would view it as not being worth much because the marginal cost to the defender of that set being held secret is only the cost that can be imposed by the tools unique to it.

If you capture someone else's tool set, then the cost to you of burning (exposing) all the other entity's tools is the intersection of that tool set with yours. The smaller the intersection, the smaller the cost. When you burn someone else's tool, you signal to them that they lost control of it. Beyond that effect, burning the other entity's tools also burns them for anyone else in the intersection set, which alerts all the actors with intersections with the set that you burned that there is some other-party intelligence that they didn't know about.

Earlier, we spoke of the defender whose job is to think about prevention. If the number of tools that are common to many actors' tool sets is a large fraction of all such tools, that is to say that the joint intersection in the Venn diagram contains the greater share of all known tools, then burning them would greatly reduce the firepower available to all actors in the aggregate, including you.

As with all modeling exercises (which is what we are doing), there are assumptions. Assumptions are not bad, but the better grade of analyst will make them, get an analysis done, and then test whether the result of the analysis was or was not critically dependent on its assumptions.

Aitel and Tait's superb article [2] on the conditions under which a free-world nation-state should reveal vulnerabilities to their authors of record has especially telling conclusions in this regard, which follow from two axioms (assumptions). The first axiom is that the free world's most dangerous opponents do not have the constraints on their discovery, use, retention, and disclosure of vulnerabilities that free world nations do. The second axiom is that the vulnerabilities that are a crucial threat to the software base of one nation are different from the vulnerabilities that are a crucial threat to another. The Venn diagram for "How much of my code base is also your code base?" is not something we have in hand, but we strongly suspect that the parts that are country-unique are the greater half, and if that is the case, then it biases the equation away from disclosing vulnerabilities to vendors of code you don't use. If opponent countries are investing in home-grown software as a strategic defense, then the bias away from disclosure to their vendors only grows stronger.

It must be acknowledged that part of our problem is the rapid rate of change which "we" otherwise praise. Beginning with Ozment and Schecter's 2006 paper [3], we have known that stable code bases under stable oversight do cleanse themselves of vulnerabilities. Clark et al. have since shown in measurable ways [4] that while the goals and compromises necessary to compete in a global market have made software reuse almost compulsory, "familiarity with a codebase is a useful heuristic for determining how quickly vulnerabilities will be discovered and, consequently, that software reuse (exactly because it is already familiar to attackers) can be more harmful to software security than beneficial." The language theoretic security group [5] has indirectly shown that the closer the code is to Turing-complete, the more likely it is to be reused, i.e., the very code that has the greatest probability of being reused is the code that has the greatest probability of being rich enough in complexity to enhance exploitability. In medical care, this is called "adverse selection" (the better the care you provide, the sicker are the people who throw themselves on your mercy).

Which leads us to the—repeat, the—fundamental question with respect to vulnerabilities: are they sparse or are they dense [6]? RAND's conjecture is that "the overlap between what is found and disclosed publicly and what is found and kept privately appears to be relatively small... [which] implies that vulnerabilities may either be dense or very hard to find," to which we might add a third option, that vulnerabilities are essentially sparse but aggregate code volume is increasing so fast that there are many more vulnerabilities than there are researchers to find them. Meanwhile, the fallout from the DARPA Cyber Grand Challenge [7] may well answer the question of sparse vs. dense and thus tell us whether or not to look for vulnerabilities (they are sparse so each killing brings them closer to extinction vs. they are dense so killings have negative return on investment).

End-of-life code bases are a special case; because they remain "unimproved" (stable), every vulnerability killed decreases the number of vulnerabilities extant. As such, it would be useful to patch zero-day vulnerabilities in no-longer-maintained software, especially for code that remains in widespread use [8].

But knowing what we know now, as underscored by Aitel and Tait, the—repeat, the—central policy question is this: are the vulnerabilities that will take you down the same vulnerabilities that will take down your opponent? If they are different, then disclosing to vendors vulnerabilities in code upon which you do not rely is an act of unilateral disarmament. Releasing a vulnerability is an aggressive act if you know someone else has it—and an intelligent move.

### *References*

[1] L. Ablon and T. Bogart, "Zero Days, Thousands of Nights: The Life and Times of Zero-Day Vulnerabilities and Their Exploits": www.rand.org/content/dam/rand/pubs/research_reports/RR1700/RR1751/RAND_RR1751.pdf.

[2] D. Aitel and M. Tait, "Everything You Know About the Vulnerability Equities Process Is Wrong," Lawfare, August 18, 2016: www.lawfareblog.com/everything-you-know-about-vulnerability-equities-process-wrong.

[3] A. Ozment and S. Schechter, "Milk or Wine: Does Software Security Improve with Age?" in *Proceedings of the 15th USENIX Security Symposium (USENIX Security '06),* pp. 93–104: www.usenix.org/legacy/events/sec06/tech/full_papers/ozment/ozment.pdf.

[4] S. Clark, M. Collis, M. Blaze, J. M. Smith, "Moving Targets: Security and Rapid-Release in Firefox": seclab.upenn.edu/sandy/movingtargets_acmccs14_340.pdf.

[5] Language-theoretic security: http://langsec.org.

[6] B. Schneier, "Should U.S. Hackers Fix Cybersecurity Holes or Exploit Them?": www.theatlantic.com/technology/archive/2014/05/should-hackers-fix-cybersecurity-holes-or-exploit-them/371197.

[7] Defense Advanced Research Projects Agency, Cyber Grand Challenge, August 4, 2016: archive.darpa.mil/cybergrandchallenge.

[8] D. Geer, "On Abandonment," *IEEE Security and Privacy* (July/August 2013): geer.tinho.net/ieee/ieee.sp.geer.1307.pdf.