DAN GEER

vulnerable compliance

Milestones: The X Window System and Kerberos (1988), the first information security consulting firm on Wall Street (1992), convenor of the first academic conference on electronic commerce (1995), the "Risk Management Is Where the Money Is" speech that changed the focus of security (1998), the Presidency of USENIX Association (2000), the first call for the eclipse of authentication by accountability (2002), principal author of and spokesman for "CyberInsecurity: The Cost of Monopoly" (2003), co-founder of SecurityMetrics.Org (2004), convenor of MetriCon (2006), author of "Economics & Strategies of Data Security" (2008) and of "Cybersecurity and National Policy" (2010). Advisor variously to global firms FTC, DoJ, DoT, NAS, NSF, USSS, DHS, and five times before Congress. Six startups.

dan@geer.org

A SECURITY PROBLEM MAY BE THEORETI-cal, but when the theoretical becomes practical it is too late for prevention. This essay is not about "responsible disclosure"; its starting point is when disclosure passes the point of inevitability—the instant when the damage control phase begins, even if silently.

Working exploits are cybercrime trade goods, instruments of national policy, or both. But we are here to look at one aspect of this and one only: what to do if a vulnerability is implementation-independent. Vulnerabilities are overwhelmingly dominated by failures of implementation, but that is not our interest.

The designers of what we call the Internet wanted one thing: survivable interoperability. As a network of networks, an Internet neither requires nor expects the construction of some single mechanism under some single control, and that more than one path exists from A to B allows the Internet as we know it blithely to accept random faults, and to route around them. The sum of these two—synthesis by amalgamation plus active fault tolerance—yields survivability, with the side effect that attribution is impossible.

The interoperability goal is inherently harder as interoperability requires out-of-band pre-negotiation of what we commonly refer to as (network) protocol. That is why we have the Internet Engineering Task Force, to standardize protocols in the Internet. Reading directly from "The Tao of the IETF" [1],

> In many ways, the IETF runs on the beliefs of its participants. One of the "founding beliefs" is embodied in an early quote about the IETF from David Clark: "We reject kings, presidents and voting. We believe in rough consensus and running code." Another early quote that has become a commonly-held belief in the IETF comes from Jon Postel: "Be conservative in what you send and liberal in what you accept."

Standing on the foundation of survivability, protocol agreement is the alpha and omega of Internet governance: the "end-to-end" principle [2], which is why the Internet embeds American values and why it works. Governments want instead to embed (their) policy into the transmission fabric, to explicitly eschew an IETF-like process. If you don't want an Internet run by thugs and nannies, now is the time to make yourself heard [3].

So the Internet is composed of an uncoordinated, amalgamational synthesis with active fault tolerance, plus the minimalist coordination of standardized protocols. Standardization is tricky—too early and it kills progress, too late and it just mummifies yesterday's fish. Most longer-lived standards are recognized in place rather than designed from scratch, but the value equation is simply whether standardization at the given moment is more enabling or more disabling. And, of course, any standard has to be implementable ("working code").

I am a strong proponent of diversity of implementations, since implementation flaws are easy to make. Look at CVE [4]. Look at the upward commercial trajectory of the code analysis companies whose entire selling proposition is that implementation errors are easy to make. But the core reason for my fondness for implementation diversity is that implementation diversity quenches cascade failure, but only for implementation-dependent flaws.

There are times when a protocol proves pretty useful and becomes so ubiquitous that we had all better hope that it has no withering flaws since, with scale-up and re-application of the protocol to jobs not foreseen during the standardization phase, flaws will out. If you accept my definition of security, namely "the absence of unmitigatable surprise," then a flaw in a protocol had better be mitigatable, because a flaw in a protocol is guaranteed to be a surprise. In other words, the question on the table is what to do when vulnerability is a consequence of standards compliance, per se.

This is no idle worry. We have a history, and if that history is any guide, then we may as well expect a future little different from the past. A few examples of this phenomenon:

- Between announcement of Kerberos availability in 1988 [5] and the formal retirement of the version 4 protocol 16 years later [6], we have an example of standards compliance implying vulnerability, embodied in an open-source code base as well as an IETF RFC.
- In 2002, Oulu University in Finland found pervasive flaws in version 1 of SNMP, the Simple Network Management Protocol [7]. In this example, it was complexity that deterred the vendor and user communities from avoiding trouble in the first place.
- ASN.1 is more complex than SNMP, complex enough that building a reference implementation is daunting. Microsoft presumably wrote their own and they surely tried hard, but ASN.1 complexity was the root cause of the critical patch in February 2004 [8]. As with SNMP, the protocol standard was designed pre-implementation.
- TCP sequence number guessing was the result of a standards process that didn't think things through, was first noticed by Robert Morris in 1985, and was the target of a corrective RFC in 1996 [9]. We sometimes get lucky; imagine if sequence number guessing was trivially possible because of how the standard was specified.
- The Wired Equivalent Privacy (WEP) protocol was where complying with a standard ensured insecurity [10]. Unlike earlier examples, the time interval between the introduction of standards-based flaws and their exposure was short, yet WEP is still in use.
- Dan Kaminsky's DNS cache poisoning work [11] expanded earlier warnings [12], but because Kaminsky's discovery was a re-discovery, we know that it was solely the public disclosure of exploitability, not the public disclosure of vulnerability, that triggered response.
- Should a message be signed then encrypted or encrypted then signed? Writing in 2001, researcher Don Davis pointed out that "Every secure

e-mail protocol, old and new, has codified naive Sign & Encrypt as acceptable security practice: S/MIME, PKCS#7, PGP, OpenPGP, PEM, and MOSS all suffer from this flaw. Similarly, the secure document protocols PKCS#7, XML-Signature, and XML-Encryption suffer from the same flaw" [13]. Davis showed that only the protocol of sign-encrypt-sign is effective, yet this flaw is still present since the S/MIME & XML groups both ignored Davis, just as the PEM group had ignored Yvo Desmedt on the same subject, 10+ yrs before. The GPG people eventually came around.

- The IPSec's committee-driven rewrite to permit username/password authentication in IKE was implemented by vendors before the committee was even done, hence producing various serious MITM issues that would take years to stamp out. Steve Kent was one of the very few who understood the issues [14].
- Marsh Ray discovered a man-in-the-middle vulnerability in the TLS standard [15]. Vendors rallied a bit for this one, but the number of unfixable implementations is high. Ray's discovery only highlighted this aspect, that of unfixable implementations.

Common-mode failure due to common-mode operations is not limited to digital worlds and security protocols.

- The US and Soviet militaries discovered EMP (electromagnetic pulse) effects early on [16]; a 1962 US nuclear test over the Pacific took down parts of the Oahu power grid. The Soviets did something quite similar in Kazakhstan. The US remediated by shielding the hell out of military gear, but eventually moved away from copper. The Soviets continued using vacuum tubes for military communications, even in planes.
- In the late '60s, phone hackers figured out how to synthesize touch-tone-style switching and billing signals. AT&T got the Feds to pass tougher laws against stealing phone service, and AT&T changed their phone-line-based protocols to something more secure.

What to do? We know that many platforms go without updates. Would it be wise to have non-compliant servers and clients treated legally as an attractive nuisance? It is unfair, but if you don't fence your swimming pool, then drowned children are your fault. Is that a good enough solution to analogize new Internet rules?

If one interprets a standard as a kind of license, then perhaps standards should come with an expire-by date. Some attacks that are not possible in today's state of the world may become possible in the future and invalidate the design environment in which a standard was crafted. Marcus Ranum has been recommending the standard "expire-by" idea for a long time. (Perhaps standards bodies need an expire-by date as well, but that's another story for another day.)

Proposed US legislation [17] is said to permit the President to shut off the Internet during times of crisis, which would (1) be impossible and (2) detonate cascading failures. Besides, as Scott Borg points out [18], disrupting the Internet is always an offensive gesture. Nevertheless, perhaps the President should be able to mandate deprecation of specific protocols, though even then it would require effort like that for Y2K. (During the 1990s, Marcus Ranum suggested: "Re-code the Internet, recompile, reboot, and blame it on Y2K.")

Sooner or later, mayn't it be a good idea to force-deprecate, say, the backbone routability of FTP, SSH v1, or SQL? However attractive that idea might first seem, it's farfetched; something like 10% of Internet backbone traffic is not protocol-identifiable, which is of interest to the SIGINT crowd

and makes protocol filtering nonsensical, even ignoring one protocol encapsulating another.

This is not just legacy; we are busy manufacturing similar situations. Kelly Ziegler notes [19] the critical ratio between firmware-update size/frequency and the available bandwidth-to-device population, i.e., utilities who want to ship new power meters with multi-MByte firmware images reachable only by sub-10Kbit/sec bandwidth. Read carefully the rationale for doing this: "It's all conforming to industry standard protocols that have been tested and vetted." In other words, the meter population could be updated within a year or so, during which the attacker would have a clear field.

So, what is the constraint on update latency for something like the electric grid? Is a year good enough? For any situation, should you take the time-to-update as the independent variable in a risk calculation and ask whether your dependence on the underlying service is too great to tolerate the resulting cycle time? If a given cycle-time is intolerable, then you have two choices: make your cycle-time shorter or make your dependence smaller.

That may be the key point: the calculus of risk as the summation of protocol dependencies. Pursuant to his potential authority to modify the Internet, should the President say to federal agencies and critical infrastructure providers alike: "I order you to be able to continue to function in the absence of the Internet"? That would mean that some agencies and/or companies would have to keep their telephone-based call centers, keep their postal service-based payment acceptance, and/or to provide software updates via CD-ROM and not just over-the-Internet download, etc.

It's time to deprecate Jon Postel's dictum and to "be conservative in what you accept." It is time to plan, for example, for the side effect of cloud computing's making some things, such as an ASN.1 compilation, less diverse but more obscure. The more we converge on standardized solutions, the more we converge on common-mode failures. We need actionable ideas on what to do when that bites hard.

REFERENCES

[1] http://www.ietf.org/tao.html.

[2] D. Reed, "End-to-End Arguments: The Internet and Beyond," 2010 USENIX Security Symposium, August 13, 2010.

[3] J. Lewis, "Docile No More: The Tussle to Redefine the Internet," 2010 USENIX Security Symposium, August 11, 2010.

[4] Common Vulnerabilities and Exposures: http://cve.mitre.org/.

[5] J.G. Steiner, B.C. Neuman, and J.I. Schiller, "Kerberos: An Authentication Service for Open Network Systems," USENIX Winter Conference, February 1988.

[6] T. Yu, S. Hartman, and K. Raeburn, "The Perils of Unauthenticated Encryption: Kerberos Version 4," Network and Distributed Systems Security Symposium, February 2004.

[7] https://www.ee.oulu.fi/research/ouspg/PROTOS_Test-Suite_c06-snmpv1.

[8] Microsoft Security Bulletin MS04-007: http://www.microsoft.com/technet/security/bulletin/ms04-007.mspx.

[9] RFC 1948: http://www.faqs.org/rfcs/rfc1948.html.

[10] W.A. Arbaugh, N. Shankar, and Y.C.J. Wan, "Your 802.11 Wireless Network Has No Clothes," *IEEE Wireless Communications*, vol. 9, no. 6 (2002),

pp. 44–51; N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11," ACM SIGMobile, July 19, 2001.

[11] US-CERT, "Multiple DNS Implementations Vulnerable to Cache Poisoning," July 8, 2008: http://www.kb.cert.org/vuls/id/800113.

[12] S.M. Bellovin, "Using the Domain Name System for System Break-ins," USENIX UNIX Security Symposium, June 1995.

[13] D. Davis, "Defective Sign & Encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML": http://world.std.com/~dtd/sign_encrypt/sign_encrypt7.html.

[14] http://www.vpnc.org/ietf-ipsec/99.ipsec/msg01734.html; http://www.vpnc.org/ietf-ipsec/98.ipsec/msg01503.html.

[15] "Authentication Gap in TLS Renegotiation": http://extendedsubset.com/?p=8.

[16] Some history at http://www.emp.us.com/emp-radiation-from-nuclear-space.html.

[17] S. 3480, as found at http://www.opencongress.org/bill/111-s3480/show.

[18] S. Borg, "How Cyber Attacks Will Be Used in International Conflicts," 2010 USENIX Security Symposium, August 13, 2010.

[19] K. Ziegler, "Grid, PhD: Smart Grid, Cyber Security, and the Future of Keeping the Lights On," 2010 USENIX Security Symposium, August 13, 2010.