The evolution of information security

Adam Shostack

Before Charles Darwin wrote his most famous works, *The Origin of Species* and *The Descent of Man,* he wrote a travelogue entitled *The Voyage of the Beagle.* In it he describes his voyages through South and Central America. On his journey, he took the opportunity to document the variety of life he saw and the environments in which it existed. Those observations gave Darwin the raw material from which he was able to formulate and refine his theory of evolution.

Evolution has been called the best idea anyone ever had. That's in part because of the explanatory power it brings to biology and in part because of how well it can help us learn in other fields. Information security is one field that can make use of the theory of evolution. In this short essay, I'd like to share some thoughts on how we can document the raw material that software and information technology professionals can use to better formulate and refine their ideas around security. I'll also share some thoughts on how information security might evolve under a variety of pressures. I'll argue that those who adopt ideas from science and use the scientific method will be more successful, and more likely to pass on their ideas, than those who do not.

1. The information security environment

Information security is a relatively new field. Some of the first people to undertake systematic analysis are still working in the field. Because the field and associated degree programs are fairly recent, many of those working in information security have backgrounds or degrees in other fields. What's more, those involved in information security often have a deep curiosity about the world, leading them to learn about even more fields. Thus, we have a tremendous diversity of backgrounds, knowledge, skills, and approaches from which the information security community can draw. Between a virtual explosion of niches in which new ideas can be brought to bear, and many different organizations to test those ideas, we ought to have a natural world of mutation, experimentation, and opportunities to learn. We should be living in a golden age of information security. Yet many security experts are depressed and demoralized. Debora Plunkett, head of the NSA's Information Assurance Directorate has stated, "There's no such thing as 'secure' anymore." To put a pessimistic face on it, risks are unmeasurable, we run on hamster wheels of pain, and budgets are slashed.

In the real world, evolution has presented us with unimaginably creative solutions to problems. In the natural world, different ways of addressing problems lead to different levels of success. Advantages accumulate and less effective ways of doing things disappear. Why is evolution not working for our security practices? What's different between the natural world and information security that inhibits us from evolving our security policies, practices, and programs?

2. Inhibitors to evolution

Information security programs are obviously not organisms that pass on their genes to new programs, and so discussions of how they evolve are metaphorical. I don't want to push the metaphor too far, but we ought to be able to do better than natural organisms because we can trade information without trading genes. Additionally, we have tremendous diversity, strong pressures to change, and even the advantage of being able to borrow ideas and lessons from each other. So why aren't we doing better?

Many challenges of building and operating effective security programs are well known. They include demonstrating business value, scoping, and demonstrating *why* something *didn't* happen. Let's focus on one reason that gets less attention: secrecy. To many who come to information security from a military background, the value of secrecy is obvious: the less an attacker knows, the greater the work and risk involved in an attack. It doesn't take a military background to see that putting a red flag on top of every mine makes a minefield a lot less effective. A minefield is effective precisely because it slows down attackers who have to expose themselves to danger to find a way through it. In information security operations, however, attacks can be made from a comfy chair on the other side of the world, with the attacker having first torn apart an exact copy of your defensive system in their lab. (This contrast was first pointed out by Peter Swire.)

We know that systems are regularly penetrated. Some say that all of them are. Despite that knowledge, we persist in telling each other that we're doing okay and are secure. Although the tremendously resilient infrastructures we've built work pretty well, we can and should do better.

For example, take the problem of stack smashing buffer overflows. The problem was clearly described in the public literature as early as 1972. According to Lance Hoffman, it was well known and influenced the design of the data flags in the main processors of the Burroughs B5500. The problem was passed down repeatedly through the 1980s and 1990s, and was exploited by the Morris Internet worm and many others. It was only after Aleph One published his paper "Smashing the stack for fun and profit" in 1996 that systematic defenses began to be created. Those defenses include StackGuard, safer string handling libraries, static analysis, and the useful secrecy in operating system randomization. Until the problem was publicly discussed, there were no resources for defenses, and therefore, while the attacks evolved, the defenses were starved. The key lesson to take from this problem that has plagued the industry from 1972 (and is still present in too much legacy code) is: keeping the problem secret didn't help solve it.

The wrong forms of secrecy inhibit us from learning from each other's mistakes. When we know that system penetrations are frequent, why do we hide information about the incidents? Those of us in operational roles regularly observe operational problems. Those incidents are routinely investigated and the

results of the investigation are almost always closely held. When we hide information about system failures, we prevent ourselves from studying those failures. We restrain our scientists from emulating Darwin's study of the variations and pressures that exist. We prevent the accumulation of data; we inhibit the development of observational methods; and we prevent scientific testing of ideas.

Let's consider what scientific testing of ideas means, and then get to a discussion of what ideas we might test.

3. Defining the problem

a. What is science?

For the sake of clarity, let me compare and contrast three approaches to problem solving and learning: science, engineering, and mathematics. Mathematics obviously underpins both science and engineering, but it will be helpful to untangle them a little.

At the heart of science is the falsification of hypotheses. Let me take a moment to explain what that means. A hypothesis is an idea with some predictive power. Examples include "everything falls at the same speed" (modulo friction from the air) and "gravity bends the path of light." Both of these hypotheses allow us to predict what will happen when we act. What's more, they're testable in a decisive way. If I can produce a material that falls faster than another in a vacuum, we would learn something fundamental about gravity. Contrast this with derivation by logic, where disproof requires a complex analysis of the proof. Science has many tools which center on falsifying hypotheses: the experiment, peer review, peer replication, publication, and a shared body of results. But at the heart of all science is the falsifiable hypothesis. Science consists of testable ideas that predict behavior under a range of circumstances, the welcoming of such tests and, at its best, the welcoming of the results. For more on the idea of falsifiability, I recommend Karl Popper's *Conjectures and Refutations*.

Science also overlaps heavily with engineering. Engineering concerns making tradeoffs between a set of constraints in a way that satisfies customers and stakeholders. Engineering can involve pushing boundaries of science, such as finding a way to produce lasers with shorter wavelengths, or pushing the limits of scientific

knowledge. For example, when the original Tacoma Narrows Bridge finally buckled a little too hard, it drove new research into the aerodynamics of bridges.

The scientific approach of elimination of falsehood can be contrasted with mathematics, which constructs knowledge by logical proof. There are elements of computer security, most obviously cryptography, which rely heavily on mathematics. It does not devalue mathematics at all to note that interesting computer systems demonstrably have properties that are true but unprovable.

b. What is information security?

Information security is the assurance and reality that information systems can operate as intended in a hostile environment. We can and should usefully bring to bear techniques, lessons, and approaches from all sorts of places, but this article is about the intersection of science and security. So we can start by figuring out what sorts of things we might falsify. One easy target is the idea that you can construct a perfectly secure system. (Even what that means might be subject to endless debate, and not falsification.) Even some of the most secure systems ever developed may include flaws from certain perspectives. Readers may be able to think of examples from their own experience.

But there are other ideas that might be disproven. For example, the idea that computer systems with formal proofs of security will succeed in the marketplace can be falsified. It seems like a good idea, but in practice, such systems take an exceptionally long time to build, and the investment of resources in security proofs come at the expense of other features that buyers want more. In particular, it turns out that there are several probably false hypotheses about such computer systems:

- ⊠ Proofs of security of design relate to the security of construction.
- ⊠ Proofs of security of design or construction result in operational security.
- ⊠ Proofs of security result in more secure systems than other security investments.
- ⊠ Buyers value security above all else.

These are small examples but there are much larger opportunities to really study our activities and improve their outcomes for problems both technical and

human. As any practitioner knows, security is replete with failures, which we might use to test our ideas. Unfortunately, we rarely do so, opting instead for the cold comfort of approaches we know are likely to fail.

Why is it we choose approaches that often fail? Sometimes we don't know a better way. Other times, we feel pressure to make a decision that follows "standard practice." Yet other times, we are compelled by a policy or regulation that ignores the facts of a given case.

4. Putting it all together: A science of information security

So what ideas might we test? At the scale which the US government operates networks, almost any process can be framed as testable. Take "always keep your system up to date" or "never write down a password." Such ideas can be inserted into a sentence like "Organizations that dedicate X percent of their budget to practice Y suffer fewer incidents than those that dedicate it to practice Z."

Let me break down how we can frame this hypothesis:

1. The first choice I've made is to focus on organizations rather than individual systems. Individual systems are also interesting to study, but it may be easier to look to whole organizations.

2. The second choice is to focus on budget. Economics is always about the allocation of scarce resources. Money not spent on information security will be spent on other things, even if that's *just* returning it to shareholders or taxpayers. (As a taxpayer, I think that would be just fine.)

3. The third choice is to focus on outcomes. As I've said before, security is about outcomes, not about process (see http://newschoolsecurity. com/2009/04/security_is_about_outcome/). So rather than trying again to measure *compliance,* we look to incidents as a proxy for effectiveness. Of course, incidents are somewhat dependent on attacks being widely and evenly distributed. Fortunately, wide distribution of attacks is pretty much assured. Even distribution between various organizations is more challenging, but I'm confident that we'll learn to control for that over time.

4. The final choice is that of comparisons. We should compare our programs to those of other organizations, and to their choices of practices.

Of course, comparing one organization to another without consideration of how they differ might be a lot like comparing the outcomes of heart attacks in 40-year-olds to 80-year-olds. Good experimental design will require either that we carefully match up the organizations being compared or that we have a large set and are randomly distributing them between conditions. Which is preferable? I don't know, and I don't need to know today. Once we start evaluating outcomes and the choices that lead to them, we can see what sorts of experiments give us the most actionable information and refine them from there. We'll likely find several more testable hypotheses that are useful.

Each of the choices above can be reframed as a testable hypothesis of "does measuring this get us the results we want?" If you think the question of, "Do organizations that dedicate X percent of their budget to practice Y suffer fewer incidents than those that dedicate it to practice Z?" is interesting, then, before testing any ideas, bringing science to information security helps us ask more actionable questions.

Similarly, we can think about building outcome-oriented tests for technology. *Proof of concept* exploit code can be thought of as disproving the trivial hypothesis that, "This program has no exploitable vulnerability of class X." Since we know that programs usually have a variety of flaws associated with the languages used to construct them, we would expect many of those hypotheses to be false. Nevertheless, demonstration code can focus attention on a particular issue and help get it resolved. But we can aspire to more surprising hypotheses.

5. Next steps

Having laid out some of the challenges that face information security and some of what we will gain as we apply the scientific method, here is what we need to do to see those benefits:

1. **Robust information sharing (practices and outcomes).** We need to share information about what organizations are doing to protect their information and operations, and how those protections are working. Ideally, we will make this information widely available so that people of different backgrounds and skills can analyze it. Through robust and broad debate,
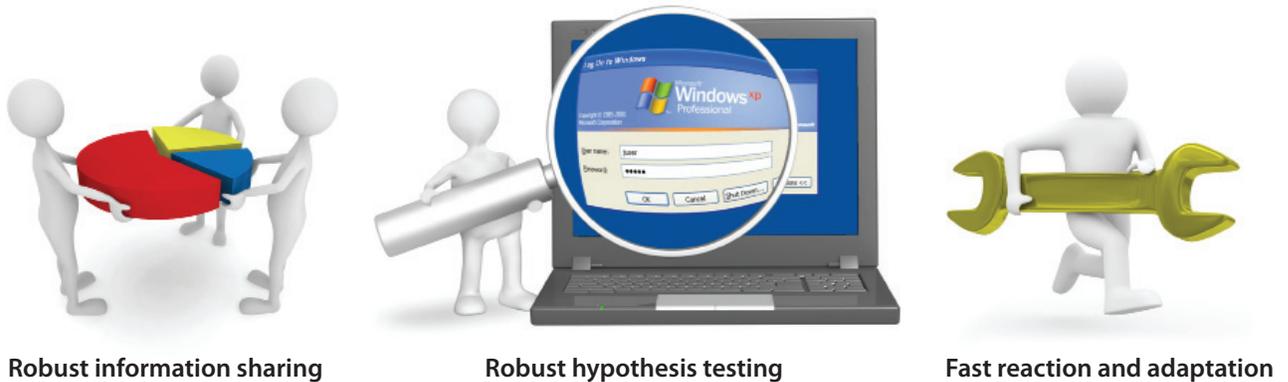
Robust information sharing   **Robust hypothesis testing**   **Fast reaction and adaptation**

we're more likely to overcome groupthink and inertia. Fortunately, the federal government already shares practice data in reports from the Office of the Inspector General and the Government Accountability Office. Outcome reporting is also available, in the form of data sent to the US Computer Emergency Readiness Team (US-CERT). The Department of Veterans Affairs publishes the information security reports it sends to Congress. Expanding on this information publication will accelerate our ability to do science.

2. **Robust hypothesis testing.** With the availability of data, we need to start testing some hypotheses. I suggest that nothing the information security community could do would make millions of people happier faster and at less risk than reducing password requirements. Testing to see if password complexity requirements have any impact on outcomes could allow many organizations to cut their help desk and password reset requirements at little cost to security.

3. **Fast reaction and adaptation.** Gunnar Peterson has pointed out that as technologies evolved from file transfer protocol (FTP) to hypertext transfer protocol (HTTP) to simple object access protocol (SOAP), security technologies have remained "firewalls and SSL." It can seem like the only static things in security are our small toolbox and our depression. We need to ensure that innovations by attackers are understood and called out in incident responses and that these innovations are matched by defenders

in ways that work for each organization and its employees.

There are objections to these ideas of data sharing and testing. Let me take on two in particular.

The first objection is "This will help attackers." But information about defensive systems is easily discovered. For example, as the DEF CON 18 Social Engineering contest made irrefutable, calling employees on the phone pretending to be the help desk reveals all sorts of information about the organization. "Training and education" were clearly not effective for those organizations. If you think your training works well, please share the details, and perhaps someone will falsify your belief. My hypothesis is that every organization of more than a few hundred people has a great deal of information on their defenses which is easily discovered. (As if attackers need help anyway.)

The second objection is that we already have information-sharing agreements. While that is true, they generally don't share enough data or share the data widely enough to enable meaningful research.

Information security is held back by our lack of shared bodies of data or even observations. Without such collections available to a broad community of research, we will continue along today's path. That's not acceptable. Time after time, the scientific approach has demonstrated effectiveness at helping us solve thorny problems. It's time to bring it to information security. The first step is better and broader sharing of information. The second step is testing our ideas with that data. The third step will be to apply those ideas that have passed the tests, and give up on the superstitions which have dogged us. When we follow Darwin and

his naturalist colleagues in documenting the variety of things we see, we will be taking an important step out of the muck and helping information security evolve. ↻

About the author

Adam Shostack is a principal program manager on the Microsoft Usable Security team in Trustworthy Computing. As part of ongoing research into classifying and quantifying how Windows machines get compromised, he recently led the drive to change Autorun functionality on pre-Win7 machines; the update has so far improved the protection of nearly 500 million machines from attack via universal serial bus (USB). Prior to Usable Security, he drove the *Security Development Lifecycle (SDL) Threat Modeling Tool* and *Elevation of Privilege: The Threat Modeling Game* as a member of the SDL core team. Before joining Microsoft, Adam was a leader of successful information security and privacy startups and helped found the Common Vulnerabilities and Exposures list, the Privacy Enhancing Technologies Symposium, and the International Financial Cryptography Association. He is coauthor of the widely acclaimed book, *The New School of Information Security.*

Further reading

Aleph One. 1996. Smashing the stack for fun and profit. *Phrack.* 1996;7(49). Available at: http://www.phrack.org/issues.html?issue=49&id=14#article

Anderson JP. Computer security technology planning study, 1972. L.G. Hanscom Field, Bedford (MA): Deputy for Command and Management Systems, HQ Electronic Systems Division (AFSC). Report No.: ESD-TR-73-51, Vol. I, NTIS AD-758 206. Available at: http://nob.cs.ucdavis.edu/history/papers/ande72a.pdf

Hoffman L. *Personal communication,* but see also the Burroughs tribute page available at: http://web.me.com/ianjoyner/Ian_Joyner/Burroughs.html

Popper K. *Conjectures and Refutations: The Growth of Scientific Knowledge.* London: Routledge; 1963. ISBN 13: 978-0-710-01966-0

Swire P. A model for when disclosure helps security: What is different about computer and network security? *Journal on Telecommunications and High Technology Law.* 2004;3(1):163–208.

Zorz Z. NSA considers its networks compromised. *Help Net Security.* 2010 Dec 17. Available at: http://www.net-security.org/secworld.php?id=10333