UC Davis
Computer Science

Title
Demythifying Cybersecurity

Permalink
https://escholarship.org/uc/item/5b10q730

Journal
IEEE Security and Privacy, 8(3)

Authors
Talbot, Edward
Frincke, Deborah
Bishop, Matt

Publication Date
2010-05-01

eScholarship.org Powered by the California Digital Library
University of California

Education

Demythifying Cybersecurity

Edward B. Talbot
Sandia National Laboratories

Deborah Frincke
Pacific Northwest National Laboratory

Matt Bishop
University of California, Davis

A large part of computer security education is tackling myths that support much of the practice in the field. By examining these myths and the underlying truths or heuristics they reflect, we learn three things. First, students and practitioners learn to separate what is empirically and theoretically supported from what is supported solely by untested anecdotes or handed-down "best practices." Second, a key part of education is the human dimension of convincing others that stories that sound right aren't proven and might in fact be wrong. They aren't necessarily wrong – but this possibility needs to be considered. Finally, we can consider myths from the perspective of teaching stories, because many evolve from activities that at one time were true or that have some accurate elements.

Modern cybersecurity measures often build on past expectations, many of which are based on outmoded assumptions. The more we fail to recognize when these assumptions don't match the new reality, the less our research will apply to the current time. The old assumptions linger on, as myths.

Cybersecurity Myths

Myths and lore often embody useful knowledge or ideas arrived at by trial and error, and codified in a way that everyone can understand and relate to. Over time, myths can change to reflect new practices and mores. As Joseph Campbell indicates, one function of a mythology is "to render a cosmology, an image of the universe," and another is "the validation and maintenance of an established order."[1]

We focus on four myths that recur in both popular literature and technical work:

1. More layers of defense are always better than fewer.
2. Running my executables on my data on my system is secure because I control my system.
3. Effective security is necessarily burdensome.
4. Trusted computing eliminates the need to trust people.

We explore them one at a time.

Myth 1: More Layers of Defense Are Always Better than Fewer

Traditionally, security's basic requirements are integrity, confidentiality, and availability, defined and balanced as dictated by the information's specific requirements. A common result is a *layered defense* or *defense in depth* strategy that protects the "crown jewels" by strictly enforced logical and physical layers of security mechanisms (the cyber equivalent of physical walls and moats).

History has shown that the layered defense is well suited for protecting physical assets. We understand their properties and how they're arranged spatially, so we know how to layer their defenses. But the layered defense has proven less and less adequate when applied to information systems' exponentially growing scale and complexity. We often don't understand how layered defenses' limitations affect their ability to protect cyber assets, nor do we understand how interactions between and across layers might reduce protection. For example, widely available technologies can bridge air gaps between cyber systems. Indeed, the IEEE 802.11 standards for wireless communication provide an example of institutionalized protocols that defeat a common layer of (physical) defense.

This problem isn't unique to cybersecurity. Even in the natural systems, such as medicine, it has become necessary to recognize that what's apparently a solid barrier might actually be porous (for example, changing beliefs regarding skin and bacteria, for example).

The nature of the layer of defense must be related to the natures of both the threats and the object being protected—and all threats are possible. For example, consider a system to which only authorized people are to have access. One layer of defense would be to place the computer in a room with brick walls and a locked door. But this doesn't secure the system when it's configured with a wireless network card; indeed, this enables the IEEE 802.11 bridging mentioned earlier. In essence, the room has become porous; thus, the walls are no longer a good defense. This suggests that understanding how complex systems interact in the face of different actions, environments, and new technologies is key to defense.

Although there are many examples in the technical arena, sometimes students enjoy looking outside the world of computing into, for example, politics, government, or business. Consider an organization with layers of people protecting a senior manager from frivolous communications (suitably defined). The layers must allow nonfrivolous messages to reach the manager. How can the people in each layer communicate with those in adjacent layers to come to an agreement on whether a message isn't frivolous, and then pass it on to the manager? And how often does something important get left out?

Myth 2: Running My Executables on My Data on My System Is Secure Because I Control My System

This myth is based on two false assumptions: you can truly control your system, and you understand precisely what security policy is implemented by the controls you select. The average person's selections won't match the expected or desired security level. Your system runs software that you've purchased, downloaded, created, or installed via an automated patching system. In the last two cases, you've used tools such as compilers and IDEs to build your software. Any of those tools can perform maliciously, without your detection.

Ken Thompson's 1984 Turing Award lecture discussed using the C compiler to insert a Trojan horse into a system.[2] In 2005, Sony BMI's digital-rights-management software, installed when users play some music CDs on Windows systems,[3] made the system more vulnerable to attackers. As Thompson concluded, "You can't trust code that you did not totally create yourself." And we might add: you can't always trust your own code.

A system's complexity and changes in the threats that can affect that system also make control a myth. How do you determine the effects of a change to a system's configuration or to the environment, at present and into the future? Configuration is based on requirements and environment, and changes to either affect system security. Combining computers and network infrastructure complicates this problem. Anyone who has worked in an organization with large networks knows that the configuration of the networks and the systems on them often don't reflect the organization's security policy. Sometimes the organization doesn't know the network's topology and the number and type of systems connected to it!

Changes to the company's organization, such as mergers with other companies, aggravate this problem. Two banks, North and South, each have a bond department and a mergers department. The two departments in North Bank make assumptions about how each other works, and the departments in South Bank make assumptions about how each other works. As each department follows its bank's security procedures, the systems work securely together. Now North and South bank merge. The bond department from North Bank needs to work with the mergers department from South Bank-but they use different assumptions. Thus, an abuser of one system can exploit this discrepancy to compromise the now-combined computer system. Understanding the nature of how combining components, organizations, and infrastructures effects their associated security policies is an important open research problem.

In this instance, this complexity gives the attackers an advantage. It provides the essential asymmetry that Sun Tsu recommends in war: the attacker need exploit only one vulnerability.[4] To be impervious, the defender must counter all vulnerabilities, known or unknown, current or that could be developed with a little ingenuity on the attacker's part. This requirement is clearly impractical.

As an example, when Microsoft developed Windows Vista, it made security the most critical attribute and emphasized that in Vista's development. The company also conducted extended beta testing. Yet within three months of release, Microsoft reported five security vulnerabilities, two of which it deemed high risk.[6] While there might have been more if Microsoft had not emphasized security, the net result is that the system was still vulnerable despite all the precautions that were taken.

Education can focus on the difficulty of assessing the actual result of combining security mechanisms. A penetration study has students analyze a system for exploitable vulnerabilities that might be used to achieve a specific, stated goal. The purpose of the exercise (as opposed to the study) is to find as many ways as possible for an attacker to achieve the goal. Phrasing this as an evaluation problem (Is the system resistant to attack, or are there enough ways to achieve the attacker's goal that the system itself is fundamentally flawed? If the latter is true, what are the flaws?) prevents the students from relying on a single exploit that works. Instead, they must study the system to fully answer the question. This shows students that, even though they might control their system, aspects of their environment might weaken system security even if their control is perfect. Another good approach might be to show how adding a layer actually provides an intruder an alternate route into a system—as in, for instance, an IDS system that contains a vulnerable system-level password enabling external system "maintenance."

Myth 3: Effective Security Is Burdensome Security

This myth holds that a system with visible, elaborately applied security measures that make using it difficult will more likely be secure than one without such "protection." Perhaps this myth's root lies in a human wish for symbols and visible protections: users, administrators, and other stakeholders sometimes feel that the system becomes secure when they are "doing something." They may feel that, as with physical barriers, if something is hard to surmount it must have a protective quality. But measures that are transparent, easy, or even fun may sometimes be far more effective than measures that require significant training, are hard to learn, or reduce productivity. The question is whether the security mechanisms actually will provide useful security enhancements or just appear to do so. But how do you tell whether a system is safer or more secure?

Without deliberately identifying some form of "security metric," security mechanisms become little more than theater driven by tradition, personality, economic pressures, or the need to "do something right away to FIX this problem!". Perhaps the biggest challenge with security theater is that the practitioners themselves often believe that the measures that they are taking are effective.

For example, Kerberos authentication is vulnerable to brute force password searches [7]. Institutions using Kerberos have attempted to address this vulnerability by constraining acceptable passwords, requiring them to be more complex; incorporating capital letters and special characters, etc. While such constraints make it less likely for an adversary to stumble upon a password using a dictionary attack, the result is that the search space for an exhaustive attack is actually reduced because these constraints exclude many otherwise acceptable passwords from the search space. This problem is exacerbated by the fact that computers capable of performing such an exhaustive attack are becoming faster and cheaper.

Consider also three-factor authentication, which involves more "visible protection" than single- or two-factor authentication. Does three-factor authentication really prevent more masqueraders from accessing the system than two-factor, or even one-factor, authentication? Does adding more factors improve security? Intuitively, at some point the inconvenience—the difficulty—of using multiple factors will make adding more infeasible. Where's the tipping point? Intuition is notoriously fallible.

All too often security means adding measure upon measure – without thinking things through. Signs of this are abundant. For instance, consider the shopping clerk who notices that a credit card does not have a signature on the back, and who then requires the customer to sign it on the back, then sign the credit card slip – and then compares the two. Is the signature comparison really helping?

Because our field is still learning how to compare the relative security of two approaches, teaching students to quantify security is no easy task. One way to enable understanding is to teach students to ask questions—such as "What data supports this?" "What is the reason you want to employ the mechanism, and how will you know if the mechanism's inclusion supports your purpose?" —and design experiments to test how people interact with security mechanisms. They'll learn how to conduct scientific experiments, interpret the data gathered, and produce results that others can reproduce. All these are critical to validating hypotheses, and intuition (and folklore) leads to hypotheses that can be validated.

Myth 4: Trusted Computing Eliminates the Need to Trust People

"If only we had no users, the system would be secure." This is the sometimes not-so-humorous plaint of the system administrator. The heart of this myth is confusion over the meaning of trust. In its usual sense, trust is a social construct that has rich overtones of ethics, morality, and religion. But computers have no moral sense and rely only on computations. A computer simply follows instructions. So, the word "trust" applied to a computer is, in reality, trust in its designers, creators, and use it. Actually, the computer is a slave to the skill, experience, and morality of the last person who got to it. If that person (and each one before) isn't trustworthy, neither is the computer.

Trust also is used in a technical sense to mean that some evidence leads you to believe that the system will meet a set of requirements. This evidence typically takes the form of analyses or mathematical proofs, all of which themselves rely on assumptions. Some of these assumptions involve people.

Two examples demonstrate this. The first springs from an examination of the methods and tools used to derive the evidence. Because people derive the evidence, the skill and understanding of these people is an important component of that evidence. Equally important are the skill and understanding of the people who developed the tools (such as theorem provers) and who validate the tools and the evidence.

The second example is operational. Suppose we have a trusted system that meets its intended set of requirements. As part of its operation, it allows only authorized users access to a set of files containing proprietary data. Who determines authorization? If Peter is authorized to access the files, and he gives his authentication credentials to Paul (or accesses the system and then lets Paul use his granted access), can the system distinguish between Peter and Paul? Currently, systems can't, so we trust users not to allow others access by sharing their credentials or relinquishing their sessions. Such environmental assumptions ought to be stated explicitly during development of the system's security requirements.

When a class discusses trusted computing, students often enjoy identifying failure points—points where the system itself trusts external information or fails to revalidate data. However, they often overlook the "people" points. Leading them to see those points emphasizes how important people are to computer security—and what happens when you overlook their role in systems.

As knowledge accumulates, we can move from myth and folklore to a more rigorous basis, in which we formalize the approach by making hypotheses and experimentally validating them. In short, we can replace myths and folklore with critical thinking and science.

The myths we've described deal with matters that can be quantified and examined critically. By teaching students to do so, we can improve the state of computer security and the quality of the education that practitioners, and indeed all students, of the topic receive.

References

1. J. Campbell, *The Masks of God, Vol. 4: Creative Mythology*, Penguin Books, 1991.
2. K. Thompson, "Reflections on Trusting Trust," *Comm. ACM*, vol. 27, no. 8, 1984, pp. 761–763.
3. M. Bishop and D.A. Frincke, "Who Owns Your Computer?" *IEEE Security and Privacy*, vol. 4, no. 2, 2006, pp. 61–63.
4. Sun Tsu, *The Art of War*, Delta Publishing, 1989.
5. M. Balmer, "When seekdir() Won't Seek to the Right Position," 3 May 2008; www.vnode.ch/fixing_seekdir.
6. R. Naraine, "90-Day Report Card: Windows Vista Fared Better than Competitors," ZDnet, 22 Mar. 2007; http://blogs.zdnet.com/security/?p=135.
7. T. Wu, "A Real-World Analysis of Kerberos Password Security," *Proceedings of the 1999 Symposium on Network and Distributed System Security* (Feb. 1999).

***Edward B. Talbot** manages the Computer and Network Security Department at Sandia National Laboratories, California. Contact him at ebtalbo@sandia.gov.*

***Deborah Frincke** is Cyber Security Chief Scientist at the Pacific Northwest National Laboratory. Contact her at deborah.frincke@pnl.gov.*

***Matt Bishop** is a professor in the Department of Computer Science at the University of California, Davis. Contact him at bishop@cs.ucdavis.edu.*

This article looks at four cybersecurity myths that recur in both popular literature

and technical work: “more layers of defense are always better than fewer,” “running my executables on my data on my system is secure because I control my system,” “effective security is burdensome,” and “trusted computing eliminates the need to trust people.”

cybersecurity, security and privacy, layered defense, defense in depth, system complexity, three-factor authentication, trusted computing, computer security, computer science education