DOI:10.1145/3630746

Article development led by acmqueue
queue.acm.org

Security must be a business enabler, not a hinderer.

BY PHIL VACHON

Security Mismatch

A "DING!" SOUNDS from your computer: You've got mail. Like Pavlov's well-trained dog, you open your email client. Something dreadful lurks at the top of your inbox: updates to a security review ticket. As you review the findings, you despair. Your team is already late on delivering features to paying customers, and you don't have the time or resources to fix everything. When the security team gets involved, the goalposts seem to move. It sometimes feels like every reviewer finds some theoretical issue they have just read a paper about and, with that, some new reason to delay your launch date. You schedule an impromptu meeting with your product management team to review the plan.

Maybe they will be flexible …

Anyone who has lived the product delivery life cycle at any mid- to large-size corporation has experienced some version of this moment—a security team gives a list of problems at the last minute, and the product team, already running late, weighs the risk of ignoring product security's guidance. Conversely, the security reviewers are brought in late in the game, give the product team a list of things to do before launch that isn't prioritized, and are unclear about how bad these issues are.

Too often, overworked product security reviewers must chuck preliminary findings over the transom, leaving it to the product team to sort out. The unconstructive feedback loop continues.

Worse, security teams eye product delivery teams warily, as if they are guilty of a mortal sin, their apparent ignorance of security best practices leaving them beyond salvation. On the other side, product delivery teams view security teams as a bunch of highly paid cowboys who cook up implausible and unrealistic risk scenarios. Product teams crave clarity about which high-priority risks to address—after all, security is exciting. It's not uncommon to see security teams fail to capitalize on this excitement, though, turning these interactions into something product teams dread. Isn't there a way to bridge the gap?

The product security reviewer breathes a sigh of relief. Another ticket done, for now, until it bounces back into the pending reviews queue. There is increasing pressure to get through more of these tickets. There isn't great guidance for the security team, either—what risks is the business, which is driving the product, willing to accept? What are even the most important products to examine for problems?

A "ding!" sounds from the security reviewer's computer. Sighing, she opens her email client. Something lurking at the top of her inbox: an all-hands update from the chief information security officer, thanking the team for all their hard work.

The "thank you" buries the lede: It is followed by a reminder the company is in a hiring freeze. But the company keeps shipping new products, so how can her team keep up with the load? Resigned, she looks at the top of the security review ticket queue: Next on deck is a ticket she has reviewed before. Maybe she will find something new …

Software is inherently complex. The economic pressures in developing software systems exacerbates this fact. The boom of frameworks, service-oriented architectures, pervasive code reuse, and

other complexity management strategies in software engineering helps reduce the "from scratch" costs.

Some might argue these strategies can take a complex design and turn it into an even more complicated problem, but a corollary of Hyrum's Law[a] is that software systems are often inadequately specified. It's in these ambiguities of specification that opportunities arise for software developers to foster their own interpretations of what is part of a public interface or a feature of a larger system, and what is not. Security teams live at this level and thrive on these ambiguities. All a product team wants to do is get something out that works for well-behaved users.

MIT's Nancy Leveson, professor of aeronautics and astronautics, reminds us often that reliability, resiliency, and security are emergent properties of a well-defined system.[b] A product is a client-facing part of a system, one that rests upon many thousands of lines of infrastructure code, all the way down to the metal and silicon of the underlying hardware. Each of these layers has its own management, reliability, and security challenges, and the development teams that own things higher up the stack are abstracted away from the details found on the lower layers. Often, security features are bolted on as an afterthought, or worse, added years after the original developers have left the company.

You can see where the mismatch occurs: Security teams exist in a very different space. They dive through abstractions, building a deep understanding of how specific components interact. They often lose track of the details that bind the layers together, however, or the fact that product teams don't even have insight into the structure of these layers. Or they may make optimistic assumptions about the complexity of the "simple" changes they ask for—your "simple interface" change could blow up into an exponential number of places.

You huddle in a conference room with your team. Your product manager, who is based at corporate headquarters on the other side of the continent, is on Zoom. The security reviewer was not clear about which problems were the worst, so you must make your best guess at priorities.

The product manager looks disinterested as you rattle off the list of findings and proposed resolutions. She interrupts and asks, "Look, is the security team stopping us from shipping?" You ponder this for a moment, filled with dread knowing what is coming next.

She sighs. "Well, ship it. We will deal with the fallout. The board cares more about revenue than what the security people say." She signs off the Zoom call. She evidently has more important things to worry about …

Because of these differences in world view, security teams tend to find themselves at odds with product teams. Even with the best of intentions, they find their guidance being ignored.

Security teams start to fall back on fearmongering to justify why their work is important to a business. This creates more friction, and an us-versus-them mindset finds fertile ground in these environments. The security team that says no to everything is a common trope in the modern corporate environment, but this response isn't given out of malice. Sometimes it's just a matter of being overwhelmed and not having good ways to answer, "Yes, but …"

Security teams must tactfully remind their partners that attacks on corporate infrastructure are lucrative for the bad guys, especially in an era of ransomware and data extortion. The sophistication of these types of attacks is only increasing, too—but criminals are not using any novel techniques to get their initial toeholds. It's amazing how weak authentication to important services, unpatched or sensitive systems being exposed to the public Internet, and social engineering are still at the root of many high-profile attacks. (Someone clever might ask, "Why don't you include 'disgruntled insiders' or '0-days' on the list?" The former is a unique business challenge to address, but it is a risk. The latter is improbable unless you are really in the wrong place and are being targeted by the right people.

One thing's for certain: Information security teams that say no must change. (Note that I never use *cybersecurity* to describe what we do; I'll tie myself into knots to avoid using the word *cyber* if I can. This is just a preference.) Hiding behind a moat makes repelling attacks easy, but bridges allow you to replenish supplies and foster relationships with customers' castles. Remember, a security team's role is to empower their business to move forward with confidence, not to hinder progress. C

IMAGERY BY MARYNOVA

a https://www.hyrumslaw.com/
b http://sunnyday.mit.edu/safety-3.pdf

Phil Vachon leads the information security program at Bloomberg's CTO's office in New York City, NY, USA. Previously, he co-founded and was CTO of a startup that built a high-speed packet capture and analytics platform and worked on spaceborne synthetic aperture radar data processing and applications.