AWS SECURITY

Creating a culture of security

Mark Schwartz
Enterprise Strategist and Evangelist
Amazon Web Services (AWS)

To be rugged is to be resilient in the face of the unexpected

It's no longer sufficient to leave security to a team of specialists who watch over the enterprise's risk posture and control it through a set of constraining policies. It's not enough to guard the boundaries of the enterprise's network with firewalls or simply implement sets of controls specified in a compliance framework. Security has become everyone's job, and its management has become a strategic concern of the enterprise. The way forward is for the enterprise to build a culture of security, an awareness of risks and controls, and a set of norms and practices that align with keeping the enterprise secure.

It's traditional at this point in an article on security to tell frightening stories of companies humbled in the face of the vulnerabilities that they left to be exploited by bad actors. I'll abstain. We are all well aware of these threats already. More importantly, we must get used to thinking of security as a positive thing, a way of building, acting, and making decisions that's just something we do naturally as builders and enterprise executives. We must treat security as part of our culture rather than reactively responding to specific threats as they're encountered.

As soon as an enterprise deploys an IT capability, innumerable attempts will be made to hack it. But the threats to our systems come not only from bad actors; IT systems can also be defeated by bad data, unexpected surges in usage, untested edge cases involving concurrent operations, cascading failures, and speed issues that multiply geometrically. In order for our systems to securely perform their jobs, they must also be scalable, resilient, available, well tested, performant, and tolerant of failures and unexpected inputs.

Security is a matter of quality

There is good news here. Security is, in the same sense that quality is, often said to be free. In the same sense that basic hygiene is (more or less) free—washing your hands, for example. In the sense that it’s cheaper to build in security rather than add it later. Security is a type of quality.

It’s about ensuring that IT capabilities will continue to work as designed when placed in real conditions—that is, under attack and facing the unexpected.

Just as there is no trade-off between quality and speed, there is no-trade off between security and speed. Interestingly, by far the great majority of exploits could be stopped by simple security hygiene. There is a very small set of weaknesses that account for the vast majority of break-ins (SQL injections and buffer overflows, for example, for those readers who are technical).

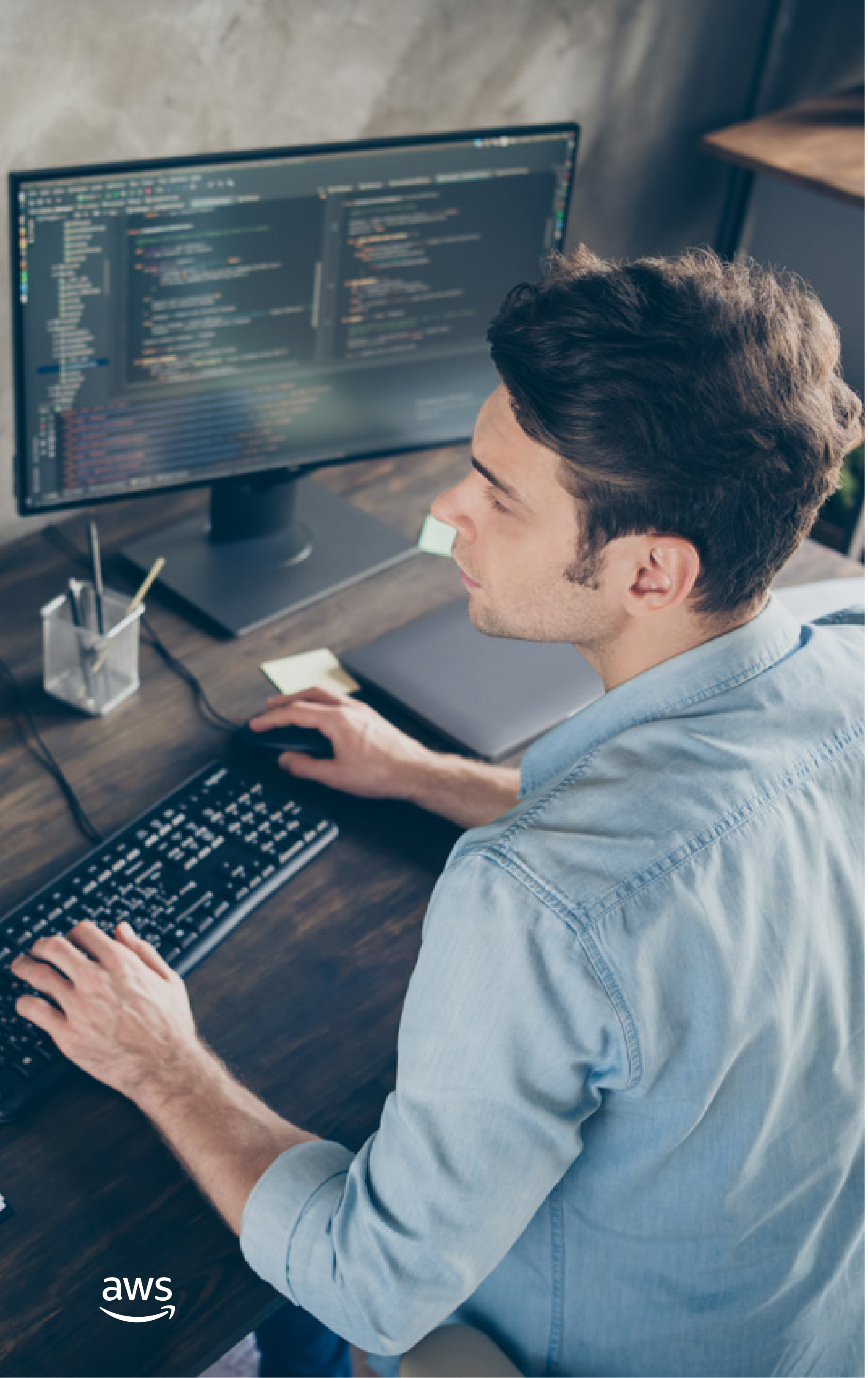
> ***“We believe that the key to producing secure code is to change your software development culture. We have to get beyond looking at the technology and look at the software development organization that created it. We believe this evolution has to start with the people, process, technology and culture of that organization.”[1]***

Ask any CISO about their greatest security fears, and you will probably hear: compromised credentials and failure to patch often enough. Add to this the top application vulnerabilities—SQL injections and cross-site scripting—and you’ve accounted for the vast majority of actual break-ins. But today’s good practices provide inexpensive ways to avoid these vulnerabilities without slowing down the delivery process or imposing undue burdens on users. Security is not fancy, geeky engineering—it’s a matter of following good practices as an everyday way of doing business. It’s a matter of hygiene.

What does a culture of security look like?

The best model that I’ve found for a hygiene-based, culture-driven approach to security is that of the Rugged Software, or Rugged DevOps movement, which advocates building secure and resilient software simply because it’s the right thing to do. According to the founders of the Rugged movement, “Rugged organizations produce rugged code designed to withstand not just today’s threat, but future challenges as well.” The key to ruggedness, they say, is cultural.

[1] “The Rugged Handbook,” Strawman Edition, August 2012



Although they talk about code and software development, these principles apply to the entire enterprise. An enterprise that treats security and resilience as a "nice to have" quality—something that only its security specialists need to worry about—as an extra cost, as a burden, or that doesn't think of it at all can never be rugged. Security and resilience should be—and are—concerns for all enterprise executives, managers, and employees. It should be fundamental to a company's culture. Culture includes those norms of the company that are reinforced by everyone's behavior and by how they view everyone else's behavior.

"

'Rugged' describes software development organizations that have a culture of rapidly evolving their ability to create available, survivable, defensible, secure, and resilient software. Rugged organizations use competition, cooperation, and experimentation to learn and improve rather than making the same mistakes over and over. Rugged organizations also actively seek out threats and create defenses before they are a problem."

Guiding principles for rugged organizations

So what does a culture of security look like? I have put together a few tenets (drawing heavily from *The Rugged Handbook*) that I think apply to a rugged organization:

Constant attacks: We understand that we are constantly under attack, both deliberate attacks and accidental ones, and build that thinking into everything we do.

Education: We value education on security (technical or non-technical, depending on our roles). We keep abreast of developing threats, accept advice from our security specialists, and seek to understand our security policies and rules.

Hygiene: Good security hygiene is part of doing things right. We do not give out our passwords. We do not share user accounts. We do not leave sensitive personal information lying on our desks when we go home at night. We use secure coding practices.

Continuous improvement: If we do leave sensitive information lying on our desks at night, we take feedback from the person who notices it, and we don't do it again.

Zero-defect approach: We do not accept any known vulnerabilities. If we discover a problem, we fix it immediately. We do not triage security defects—deciding that some are worth fixing now and others are not.

Reusable tools: We look across all our IT systems and build tools and processes that can be shared between them. These might include reusable logging and monitoring, enterprise-wide user provisioning, and standardized onboarding and offboarding processes for employees.

Unified team: All parts of our organization collaborate to make security strong and systems resilient.

Testing: We test our systems rigorously (primarily with automated tests) while they are being developed and also while they are in production. We test failure scenarios and our ability to respond to them.

Threat modeling: We try to think like a bad guy, just as we try to get inside our customers' minds. We brainstorm possible routes an attacker might take to defeat our controls and test to make sure they cannot.

Peer reviews: Each technologist should think both about possible defects in their work and possible security vulnerabilities. Code should always be reviewed by a peer, who should also be looking for vulnerabilities.

> ***"The speed at which data is available dictates the speed at which decisions can be made."***
>
> ***Mark Schwartz, Enterprise Strategist and Evanelist, Amazon Web Services***

> *"When security is treated as an essential aspect of a mission (or business) accomplishment, there are rarely as many trade-offs involved as people seem to believe."*
>
> *Mark Schwartz, Enterprise Strategist and Evanelist, Amazon Web Services*

How we build a culture of security at USCIS

We thought a lot about security when I was CIO at the United States Citizenship and Immigration Services (USCIS), an agency in the United States Department of Homeland Security. But when I first joined, security was not part of everyone's everyday processes. Sure, we required all employees to pass an annual training course on security awareness. We had extremely skillful security engineers and penetration testers dedicated to keeping our systems safe. And we periodically performed social engineering audits.

But for most employees, security was seen as a burden. Developers saw security as a delay in getting their code deployed to production. "Secure" meant satisfying security testers and passing compliance requirements. Systems were launched with known vulnerabilities that were listed in a tracking system to be addressed later. Allowing these vulnerabilities to exist was called "a business decision" based on "an analysis of risk."

Changing the perspective

As CIO, I was the authorizing official, the person who decided if each system was secure enough to be launched. I did this by granting an Authority to Operate (ATO) after consulting with my CISO and security team. The government ATO process was deliberately designed to give agencies flexibility: As a business executive, I could make risk-based trade-offs to ensure that security was treated in a way that was practical and that balanced security goals with mission accomplishment.

Unfortunately, this approach conveys the misleading idea that security is opposed to mission accomplishment and that trade-offs must be made. But when security is treated as an essential aspect of a mission (or business) accomplishment, there are rarely as many trade-offs involved as people seem to believe.

It's important to note that these five mechanisms do not involve substantial spending or require much time once they are established. They are not about trade-offs between security and product delivery. I'll dig deeper into each of these five mechanisms below.

The approach we took to building a culture of security at USCIS looked like this:

Consistently connect security to mission objectives.

Build security into everything and correct mistakes quickly.

Establish norms and high standards for security hygiene.

Adopt a zero-defect approach.

Continuously vet security in development and production.

Consistently connect security to mission objectives.

We all have a responsibility to our customers and other stakeholders to maintain the security of our systems. Customers trust us with their personal data. Shareholders trust us with the company's financial data. In the case of USCIS, the public trusted us to maintain the integrity of the country's immigration system. These responsibilities rest with everyone in the organization, from leaders to individual contributors.

The CFO should be clear that security matters for financial well-being. The CMO and head of sales should be clear that only if the company protects the integrity of its systems can they deliver on their implicit or explicit promises to customers. The COO should understand that operations include operating the company in a reliable way, with integrity and consistency.

Everyone within the organization should view security as a critical job requirement. Everyone in the company should answer these questions for themselves: "Is it OK for us to let our systems be compromised? For our applicants' data to be stolen? For a denial of service attack to make us stop providing services?" If you don't believe security is important, then you are misunderstanding your job.

Instead of meeting with only my security folks to decide whether to grant an ATO, I insisted on a meeting with all key stakeholders, including business sponsors, product owners, and development teams. I asked questions to determine the security posture of the system and made sure everyone could see and understand any issues that were exposed. I required people to commit to actions they said they would take to improve security. And I supported the security team in their findings and made it clear to everyone that security was nonnegotiable.

Build security into everything and correct mistakes quickly.

Our periodic audits revealed the bad news: Employees were being fooled by social engineering attacks. No matter how much we trained them, employees would give up their passwords when asked by "a technician from the help desk." A dedicated adversary could design spear-phishing attacks that would fool people into clicking on links. People will write down their passwords because if we make passwords strong enough to be effective and insist that they be different for every system, then they are too hard to remember.

That's why we moved entirely to multi-factor authentication. It's not a panacea, but it can make life much less difficult. We introduced automated security tests into our development pipeline that gave developers immediate feedback if they introduced a common security vulnerability—and showed them what the vulnerability was and how to avoid it. We created reusable code that implemented good security practices (identity and credential management, auditing and logging, and so on) and could be included easily in new systems. We installed encryption software on everyone's laptops.

When our penetration testers found a vulnerability, we gathered everyone together and had the testers present their findings so we could all understand how the tester had fooled us and how we could avoid a similar incident in the future.

Establish norms and high standards for security hygiene.

You wash your hands when leaving the restroom. If you are a developer, you validate your input. Washing your hands makes it less likely you will ingest germs. Validating your inputs makes it less likely you will succumb to an SQL injection hack or a buffer overflow. It's just something you do, and if you don't, you become a social pariah.

Washing your hands will not keep you safe from hereditary neurological diseases, but the most common illnesses are not hereditary. Similarly, there are complex hacks that ordinary hygiene will not keep you safe from, but by far the great majority of hacks exploit the simple things we forget to do or the careless mistakes we make.

Good security hygiene is virtually free and extremely effective. It's mostly a matter of building new habits, but it prevents the vast majority of everyday security exploits. Sensitive documents should be shredded. User accounts should be given the minimum privileges possible. When employees leave the company, their accounts should be immediately terminated.

> “
> ***Good security hygiene is virtually free and extremely effective. It is mostly a matter of building new habits.”***
>
> ***Mark Schwartz, Enterprise Strategist and Evanelist, Amazon Web Services***

Adopt a zero-defect approach.

I don't understand the idea of allowing a known security vulnerability into production. What good does it do to bolt most of your doors, but leave one swinging open? It only takes one door for a thief to get into the house. This is not a risk/cost trade-off. You have wasted your other security spend if you leave a door open.

In a continuous delivery environment, code must pass all tests before it can be promoted. It's a simple green light/red light condition. All security should be treated this way. This might seem like a tremendous burden. But except in the case of a legacy system that has many outstanding security defects, good practices with automated regression test suites guarantee that, at any given moment, the only place there can be a defect is in the small amount of code that was just checked in.

At USCIS, when I reviewed each legacy system (with all of the major stakeholders in the room), I would ask about the known vulnerabilities there were. I insisted on a plan for remediating each of them or establishing compensating controls and a very aggressive timeline for doing so. All stakeholders came to understand that known vulnerabilities were not acceptable.

Continuously vet security in development and production.

The process at USCIS involved reviewing each system every two to three years to make sure it was still secure and then issuing it a new ATO. Instead, we started enrolling each system into an ongoing authorization process, where the system was continuously tested and assessed by automated tools while it was running.

If any vulnerability was found, there was an immediate escalation process to deal with it. Essentially, we extended our pre-release security testing process into post-release; the same things that would have prevented an ATO when the system was first launched would now trigger an immediate escalation and remediation after launch as well.

This made security a proactive matter rather than something we thought about only when forced to. We not only wanted to release secure systems, but we also wanted to prove to ourselves at every instant that we were as secure as we could be. Another way to look at it is that there's urgency not just when a system is under attack but also when a flaw is detected that might make it vulnerable to attack.

Build your own culture by creating new habits

The techniques described here all contributed to establishing a culture where security was valuable and considered something we owed our stakeholders. The techniques involved breaking old habits and creating new ones, but they were not expensive or time-consuming. And most of all, they helped do away with the problematic idea that there is a trade-off between being secure and satisfying customers.

Related content

Cultivating Security Leadership

Hear how enterprise CISOs are investing in their people to safeguard their organizations.

AWS Security and Compliance Quick Reference Guide

Learn how you can achieve savings and scalability while maintaining robust security and regulatory compliance.

AWS re:Invent 2019 AWS Security Leadership Session

Stephen Schmidt, AWS chief information security officer, shares his perspective on the current state of cloud security.

Innovative leaders share how they drive business growth and transformation.

Learn more ›

About the author

Mark Schwartz, Enterprise Strategist and Evangelist, Amazon Web Services

Mark is the author of War and Peace and IT: Business Leadership, Technology, and Success in the Digital Age, The Art of Business Value, and A Seat at the Table: IT Leadership in the Age of Agility.

Before joining AWS, he was the CIO of the United States Citizenship and Immigration Service (an agency in the United States Department of Homeland Security), CIO of Intrax, and CEO of Auctiva. He has an MBA from Wharton, a BS in Computer Science from Yale, and an MA in Philosophy from Yale. Mark is also an avid blogger.

Learn more about Mark ›