

Meet The Product Security Leaders to Watch in 2026

Product Security is evolving at the speed of AI. Read insights from the leaders securing the next wave of innovation and learn how they're navigating this new era of product security.



The End of Business as Usual

Product security has changed more in the last year than it has in the previous decade.

Organizations are now dealing with three forces at once: accelerated development, an expanding and increasingly fragmented attack surface, and autonomous systems that can both generate and exploit code. The result is a new risk profile entirely, one where the threats are AI-generated, the surface is always expanding, and human-speed responses simply don't keep up.

Security teams know the math doesn't work. Traditional tools and processes weren't built for AI-driven development, and adding more of them only deepens the problem. More tools, more alerts, more manual triage creates noise without improving outcomes.

Some teams are already using AI to fight AI — but layering AI tools onto a broken operating model isn't enough.

What’s required is a fundamentally different approach, and to understand what that looks like in practice, we spoke with leaders operating at the front lines of this shift.

We asked our 2026 cohort of Product Security All-Stars questions about how AI is reshaping both attacker and defender capabilities, which long-standing “best practices” are breaking under this new pressure, and what it actually takes to prioritize and reduce real, exploitable risk at scale.

Their answers point toward a new model: one where security is continuous, context-driven, and capable of operating at the same speed and scale as AI-driven development. This is the foundation of the Agentic Development Lifecycle (ADLC).

Table of Contents

Introduction: The End of Business as Usual 02

✦ **Nikola Dalcekovic,** Cybersecurity Officer, Prosumer Software| 2X All-Star 04

✦ **Priya Balasubramaniam,** Head of Innovation Center, Product & Application Security 06

✦ **Anshuman Bhartiya,** AI AppSec Engineer 08

✦ **Miriam Celi,** Director, Security Architecture 10

✦ **Julie Chickillo,** VP, Head of Cybersecurity 13

✦ **Jamie Dicken,** Director of Security Platforms and Architecture 15

✦ **Daniel Hammon,** Director, Information Security and Compliance 17

✦ **Conleth Kennedy,** Manager, IT Application Security 19

✦ **Kevin Markley,** Senior Manager of Platform Security 21

✦ **Kimberly Mattheys,** Product Security Leader 23

✦ **Cássio Pereira,** Senior Application Security Engineer 26

✦ **Rusty Perry,** Enterprise CISO Leader 27

✦ **Chase Pettet,** Principal Security Engineer 29

✦ **Michael Westphal,** DevSecOps Lead 31

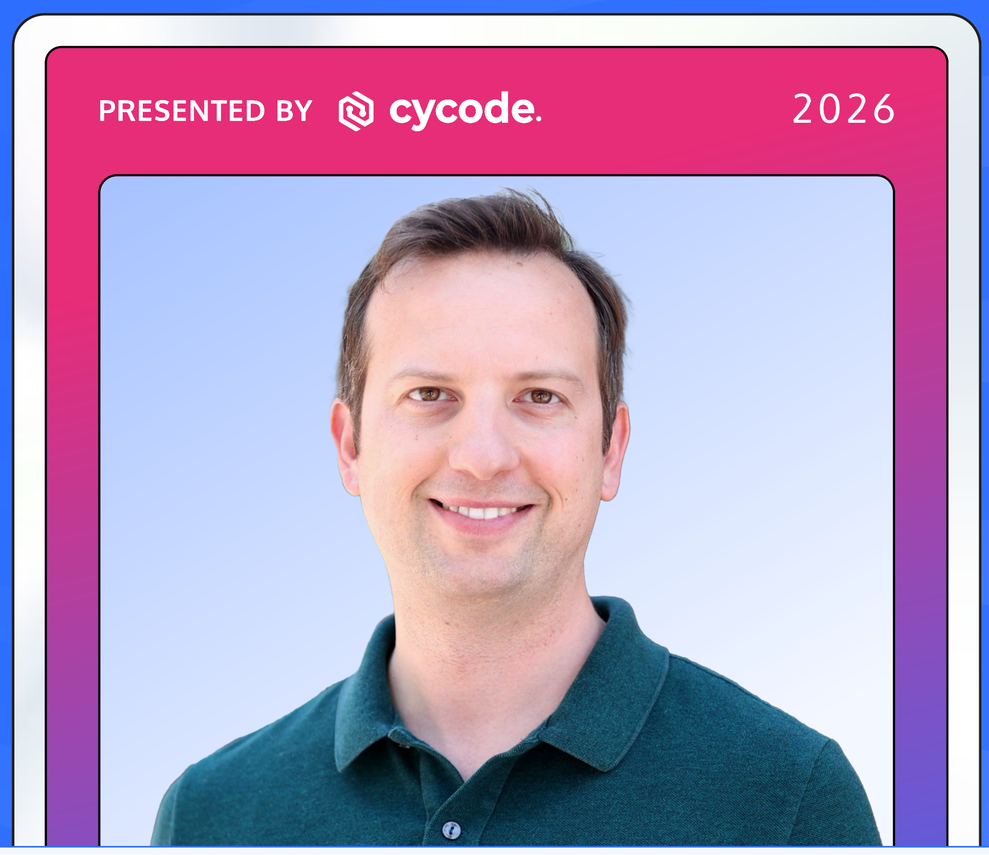
Conclusion: Closing the Security Gap 33

2X ALL-STAR • SINCE 2025

Nikola Dalcekovic

Cybersecurity Officer, Prosumer Software

Schneider Electric



Bio

Nikola Dalcekovic has over a decade of experience in software and security engineering. He began his career as a software engineer before transitioning to product management, where he established the Product Security Office to enhance secure practices across a workforce of over 500 engineers. Over the past eight years at Schneider Electric, his team has scaled globally, supporting multiple high-risk profile products across three different R&D regions.

In 2022, Nikola relocated to California to pursue innovation in the industry. He holds a PhD in the security of multi-tenant systems, has published over 10 scientific papers, and served as a professor at the University of Novi Sad, Serbia. Nikola is certified in CSSLP and CISSP, and has completed an INSEAD course in business leadership and strategy with distinction.

Currently, Nikola serves as Cybersecurity Officer (CSO) for Prosumer Software at Schneider Electric, where he provides strategic cybersecurity leadership by partnering closely with executive leadership to strengthen product and data security, guide risk and incident readiness, and align secure development practices with the division's operational, commercial, and regulatory priorities.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> The one book that really stayed with me (even though it's not a security book) is Flow. I read it maybe 20 years ago. It's more about philosophy, about how people find their own flow so they can enjoy life and be productive.
>
> I always relate that back to developers. Developers need their own flow to be productive, and security has to fit into that, not break it.
>
> More recently, I read How to Measure Anything in Cybersecurity Risk. That one is much more practical from a cybersecurity perspective. It's very useful for areas like insurance, where you really need to quantify risk properly, but it also broadens your thinking.
>
> It helps you start putting dollar amounts behind risk and expressing findings in business-relevant language. That's something we don't always do well as an industry.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> I remember situations where I supported shifting security discussions to the right (postponing them) because we had tight deadlines. The intention was good. We wanted to move fast, get to market, and increase revenue.
>
> But that's a mistake. If you catch a design-level issue later, you're in a position where you either accept the risk or you fix it at a massive cost, and it delays you even more than if you had addressed it early. I've seen that happen.
>
> That's where I changed how I look at product security. It's not just part of a development lifecycle where it can be perceived as a checklist, but part of the strategic discussion. It helps you get to the finish line properly and safely.
>
> When you're at the table with business teams, security often looks like an extra requirement, another meeting, and another review. But it has to be positioned as an integral part of what we're trying to achieve as an organization. That's when it becomes a real strategic lever.

Nikola Dalcekovic, Cybersecurity Officer, Prosumer Software

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> What looks like a clean, secure system on the surface may actually be powered by a web of AI components that aren't fully governed or understood. Traditional metrics might say everything is fine, but they don't capture the risk introduced by those interactions.
>
> That really shifts the discussion. You now have to consider things like runtime manipulation of those models, or how they interact with the product as a whole.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> Honestly, I don't know what to expect in five years. The pace of progress is so fast that it's hard to even imagine. You hear people talking about AGI in a couple of years, and from there, who knows.
>
> What I do think is that over-reliance on AI agents is a real risk. Even the idea of "human in the loop" has limitations. If AI gets something right 50 times, how focused are you really going to be the 51st time? It's human nature. So just saying "we'll put a human in the loop" isn't enough.
>
> If we get it right, I think it comes down to strong quality frameworks: automated testing, proper guardrails, and a structured way of integrating AI into development. If we don't, we end up with a sprawl of unmanaged risk.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> The idea that we need to fix all vulnerabilities. I think that's wrong, especially now. The number of vulnerabilities is only going to increase. With AI, exploitation becomes faster, cheaper, and more accessible.
>
> So the idea of having a product with zero vulnerabilities isn't realistic, and in many cases, it's not even the right goal. Context matters. Without it, just listing vulnerabilities or disclosing everything can be misleading. Even with regulations pushing for full transparency, without proper context, that information doesn't always help.
>
> Trying to fix everything can also be a poor use of effort and investment. We need to move toward understanding which vulnerabilities actually matter in a given context, and prioritize accordingly.

Priya Balasubramaniam

Head of Innovation Center,
Product & Application Security

ciena



Bio

With over 20 years of experience in the technology industry, Priya is a seasoned security leader who has successfully transitioned from delivering complex optical networking solutions to championing secure software development practices across product engineering. As the Head of Innovation for Product & Application Security, she is a strong advocate for building security into the DNA of products, while proactively exploring emerging technologies and piloting cutting edge solutions to reduce risk and minimize attack vectors across the modern software supply chain.

Beyond her technical leadership, Priya is a committed mentor and active member of Women in Tech resource groups, where she focuses on enabling the personal and professional development of women. She brings a deep commitment to customer transparency and trust and is passionate about elevating product security as a strategic business enabler, not just a technical function.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> My perspective really shifted during my time in software development, building complex optical networking solutions for major global telecom providers. Their core concern was always the security of the network.
>
> They needed absolute confidence that sensitive data was traveling through a system with the utmost safety. That customer expectation, combined with the principles of Secure by Design and Secure by Default, changed everything for me.
>
> Security cannot be an afterthought. It has to be embedded as early as possible. Things like authentication, encryption, and data validation should be part of the product's DNA. They shouldn't be something a user has to opt into or configure later.
>
> That shift (seeing security as a built-in, resilient standard for critical infrastructure) is what led me to move from software development into a dedicated security role.

What is the best piece of advice you have ever been given that you still apply to your career today?

> "Your actions speak louder than your words." This is something I live by and pass on to my team. You build credibility through results and outcomes. Demonstrating behavior has far more impact than just talking about it.
>
> I've seen that play out in very real ways. I'm an active member of our "Women at Ciena" employee resource group, and I volunteer regularly. What surprised me was seeing my male direct reports start taking their own initiatives to support underprivileged women and kids in their communities.
>
> I never told them they had to do it. They saw the action and chose to contribute. That's how trust is built: through what you do.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

High-profile supply chain breaches like SolarWinds and Log4j really shifted the industry's mindset. They exposed how deeply embedded third-party and open source components are in our environments. A single vulnerability can create a massive ripple effect across an entire ecosystem.

Internally, that drove us to elevate product security as a strategic priority. We focused on gaining full visibility into our "software ingredients" — every dependency, every nested library. We also implemented programs like PSIRT, where we proactively and transparently inform customers about known vulnerabilities and what actions they can take.

Being responsive and transparent builds trust. That's where security becomes a business enabler.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

The intentional poisoning of the code supply. This is where malicious logic is added into libraries or dependencies in a way that evades detection and stays dormant for months until it's triggered. It doesn't get attention because it's invisible. It doesn't break functionality. It doesn't disrupt integrations. There's no immediate alarm going off.

A lot of boards believe that if they fix all known CVEs, they're covered. But malicious code doesn't always come with a registered CVE. That's where the blind spots are. These are risks that go beyond basic scanning. You need deeper visibility and a different way of thinking about trust in your software supply chain.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

If we get it right, AI becomes a trusted teammate. It acts as both a guardian and a guide throughout the development lifecycle.

It doesn't just write code. It identifies vulnerabilities early and suggests remediation before code is even committed. You start to see predictive threat modeling: AI looking at emerging attack patterns and hardening applications in advance.

In that kind of environment, breaches caused by known vulnerabilities become very rare, because systems are continuously monitored and self-correcting. If we get it wrong, it becomes a nightmare.

We'll create security debt at scale because we're producing insecure code faster than ever. Teams may start accepting AI-generated fixes without proper validation, which can introduce new problems or expand blind spots.

Instead of reducing risk, we increase the attack surface and erode customer trust. The outcome depends entirely on how we use it.

Anshuman Bhartiya

AI AppSec Engineer



Bio

Anshuman Bhartiya is a technical security leader and researcher with over a decade of experience building and breaking software at scale. Currently, as the AppSec Tech Lead at Lyft, Anshuman specializes in developing automated security programs that keep pace with high-velocity engineering teams. A builder at heart, he is a vocal advocate for a "bias to action," believing that security engineers must deeply understand how software is built to effectively secure it.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> Lately I've listened to a few podcasts where Peter Steinberger, the creator of OpenClaw, came as a guest. He describes how he has been building OSS tools including OpenClaw, where he has multiple Codex sessions open and he is essentially acting as an orchestrator guiding the coding agents and letting them build things for him. He doesn't review each and every line of code the coding agents generate, but he understands at an abstract level what's being built and shipped.
>
> Up until now, we've approached building and shipping software in a very specific way, but this represents something fundamentally different, something security engineers aren't fully prepared for yet. Humans and security engineers reviewing code are a thing of the past. We need to embrace the new reality and figure out innovative solutions of letting AI review AI generated code—with some human involvement—in a way that it doesn't necessarily become a bottleneck, but more like an assurance/verification layer.
>
> It made me realize we're moving toward a model where humans aren't just guiding AI, but co-creating with it. And that's a powerful shift.

What is the best piece of advice you have ever been given that you still apply to your career today?

> "Bias to Action". When you're faced with a challenge or something difficult, instead of just talking about it, go build something. Create a prototype, test an idea, try to solve the problem. That process fuels the curiosity that's essential in security. You need to understand how things are built if you want to truly understand how to break them, or fix them.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> The silent threat is the sheer power of these newer models from the frontier AI labs. We're already seeing cases where models like Claude Mythos can identify vulnerabilities that have existed for decades without being noticed in open sourced repos. With every model release, it seems like the frontier labs are coming after security startups and their moats.
>
> If the moats don't involve any form of data analysis or some form of intelligence layer on top of customers data, it's going to start eating away at the product layer for a lot of early stage startups who are still trying to hone their problem statement and product market fit.

Anshuman Bhartiya, AI AppSec Engineer

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> Engineering teams these days are looking to bake in AI in their processes, tools and SDLC. Everyone is building MCP servers, skills and powerful agentic applications that have the capability to run autonomously for a long time, making decisions without a lot of human involvement.
>
> The fundamental way of building and shipping software is changing on a daily basis. The status quo needs to be challenged and rightly so. In such cases, moving fast is important but doing so without introducing huge security debt and risks is even more important as we are dealing with systems that are capable of being smarter than humans.
>
> All this to say that Product Security is going through a transition where enabling teams to safely adopt AI can be a great OKR to have. Doing so would enable businesses as opposed to being a technical hurdle. ProdSec teams need to upskill and uplevel supporting orgs on their AI adoption journeys.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> If we get it right, the line between AppSec and engineering essentially disappears. Everyone becomes an AI-enabled builder in some way, and security shifts from being reactive to being truly secure by design and more proactive.
>
> You're embedding requirements into AI agents early, so those guardrails are built in from the start with verifications happening at multiple layers in the modern, AI-native SDLC.
>
> If we get it wrong, we're just chasing our tails. The speed of AI agents will far outpace anything humans can keep up with, and we'll always be reacting instead of staying ahead.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> Manual, ad-hoc threat modeling is becoming an anti-pattern. It's something we've relied on for a long time, but it doesn't scale. It's too resource-intensive and too slow. It's also often not continuous. We need to rethink it as a continuous, automated process, something that evolves alongside the system, rather than a one-off adhoc exercise.
>
> The manual threat modeling needs to be replaced by AI doing the threat modeling continuously, and providing more context aware and org specific outcomes. It's also super important to create an iterative feedback loop where the findings from threat modeling are fed back into the system so that it self learns and improves over time.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

> It excites me. I think it fundamentally changes the role of the security engineer. Instead of doing manual tasks, you become more of an orchestrator: managing a fleet of agents and ensuring they have the right context and are making the right decisions. It's a very different mindset, but a very exciting direction.
>
> It also means that the humans on my team can now spend their time and energy on more meaningful and complex tasks like architecting and designing secure by default systems, and less on things like triaging low hanging fruits.

Miriam Celi

Director, Security Architecture

Humana



Bio

Miriam is a seasoned software security leader with an impressive background of over 20 years in the IT industry, with more than a decade dedicated to application and product security. She is committed to fostering a strong culture of security through collaboration, mentorship, and advocacy for customer trust.

Throughout her career, Miriam has successfully led and evolved application security programs, provided invaluable advice to development teams on secure architecture and design, and spearheaded numerous initiatives across threat modeling, DevSecOps, security testing, and risk management. Her expertise spans a diverse range of industries, including financial services, healthcare, identity management, and enterprise software.

Miriam is a CISSP, CCSP, and CSSLP-certified security professional and holds a Master of Science in Computer Science degree from Florida Atlantic University and a Master of Business Administration degree from Nova Southeastern University. She is a named inventor on U.S. patents in the areas of multimodal application development, proxy insertion, and enhanced authentication methods using dynamic geographical location data. Her background reflects her long-standing focus on innovation, security, and scalable software design. She is also known as a security evangelist who enjoys mentoring others and translating complex security concepts into practical, actionable guidance for engineering teams and leadership alike.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

One book that has fundamentally changed how I think about security has got to be Adam Shostack's Threat Modeling: Designing for Security. This is one of the first books I got when I started my journey with threat modeling, and the book content holds up to this day. I had the opportunity to meet him in person giving a presentation on threat modeling at a past Black Hat conference and was able to get him to sign my book. It was an amazing experience for me.

What I like about this book is that it's very practical and anyone can easily apply it when building software. The book introduces four main questions that will help you to threat model:

What are we building?
What could go wrong?
What are we going to do about it? And
Did we do a good job?

The book also helps you to think like an attacker in a structured way but also from a defensive perspective. You're always going back: Did we do a good job? Did we address the high-priority threats first?

So, for anyone involved in security architecture, threat modeling, or even leading teams that build products, this book is essential. Even after years in security architecture, application security, and leadership, I still refer back to it.

What is the best piece of advice you have ever been given that you still apply to your career today?

> Speak what's in your mind and also to listen and not be afraid to share your opinion. Early in my career as a young woman in a male dominated field, I was very hesitant to speak up. But I eventually realized that people really want to hear your thoughts and ideas, and that everyone's perspective is valuable. Holding it back doesn't help anyone, especially yourself. Over time, I was able to overcome that fear, and now I feel more empowered to say what's on my mind and share my point of view.
>
> I've realized that by sharing your perspective, you can prevent misunderstandings and improve the outcomes of the team and the organization. However, it's just as important to listen as well and not just be heard, as listening to others can help you gain a better understanding of other areas or concerns you may not be aware of.

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> The single most important non-technical skill would be translating security risk into business language. This is really hard for us security professionals and technologists. Most of us get into the weeds: vulnerabilities, severity ratings, technical jargon. But if you can't explain the impact to executives in terms of business outcomes, the risks won't get prioritized and won't get addressed.
>
> I find this similar to when giving a presentation in that you have to know your audience to get the right message across showing how the risk can affect customers, compliance, and the company. If you're too deep in technical detail, executives won't understand the risk that you are trying to communicate and your message will be lost and won't get prioritized. This also applies to communicating risk to engineering and product teams and other stakeholders to understand the "why" behind the security requests.
>
> This is a critical skill to practice as a security professional, and I always try to be mindful of the audience when communicating security risks.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> Trying to improve security engagement earlier in the software development lifecycle (SDLC) proved that I needed to use a more lightweight version of threat modeling then what I originally was attempting to do. I found out that full system threat models are not very conducive to agile engineering practices and are difficult to plug in earlier in the SDLC (shift left) so that was generally brought in too late in the workflow and resulted in issues that were already embedded in the design and more difficult to resolve. This created friction with engineering and product teams and caused delays in releases.
>
> So, I experimented with a more lightweight threat modeling approach centering back on the initial four main question framework and that helped make a big difference in adding that earlier in the SDLC resulting in security issues being identified before development began, reducing rework and remediation efforts, and resulting in improved product security.
>
> This experience made it clear to me that product security is not just about finding and fixing vulnerabilities; it's about shaping and influencing how products are built with security by design and sometimes that means modifying your approach to align with engineering practices.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> I think a silent threat at the product layer that isn't getting enough attention is the implicit trust within software ecosystems (third party services, APIs, open source, CI/CD pipelines). Organizations nowadays fully rely on and trust these software ecosystems which attackers can compromise and propagate that access to leverage further attacks on many unsuspecting organizations. The attack surface is no longer the application itself but now also the entire supply chain around it.
>
> That means we need to think beyond vulnerability scanning and consider the software supply chain integrity, dependency governance, CI/CD pipeline security, and runtime trust boundaries. Supply chain risk needs to be communicated at the boardroom level, so it's understood and accounted for. Organizations that prepare themselves for these types of attacks, will be in a better position to respond and recover quickly from such situations which are now inevitable.

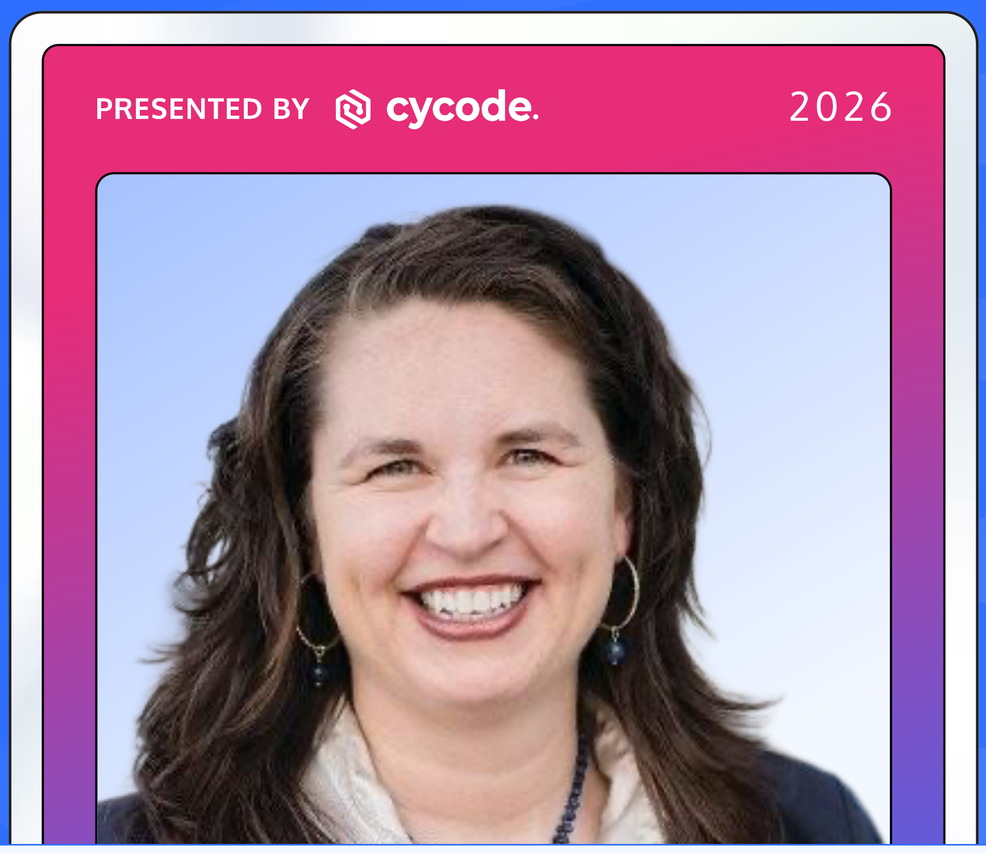
Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> If we do it right, I see AI helping with threat modeling among other things. It can also help make security more accessible, helping teams work faster. At the same time, I think it's still important that human oversight is in the loop as AI generated output and actions may not always be accurate or correct.
>
> The same goes for AI generated code which can be riddled with vulnerabilities and may not always be the simplest, most optimized solution. Security teams will need to position themselves more on the governance side of AI than just the oversight or reviewers of AI generated code. They will also need to pay attention to the AI ecosystem to ensure adequate controls or guardrails are included in every step of the SDLC.

Julie Chickillo

VP, Head of Cybersecurity

Guild



Bio

Julie Chickillo is a veteran security leader with over 15 years of experience across governance, risk, compliance, and DevSecOps. Currently the VP and CISO at Guild, Julie is known for her pragmatic approach to security—transforming it from the traditional "big book of no" into a frictionless business enabler.

An early adopter of modern cloud-native practices, Julie has successfully implemented fully automated DevSecOps programs and scalable security data lakes. She is a vocal advocate for the "agentic shift," focusing her strategy on moving security teams from manual triage to high-level orchestration where humans provide the critical context and oversight for AI-driven systems.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up?

> The thing that really changed how I think about security was The Phoenix Project. I read it when it first came out because our whole tech team was told to go through it.
>
> What stood out to me was that it wasn't really a security book at all, it was more of a philosophy book. But it completely changed how I approached my role.
>
> What still holds up today is the idea that everyone has to change how they think in order for the company to move forward. Security can't just be the "department of no" or the audit team dropping a list of problems. It has to be about enabling teams to move faster in a productive way. Especially now with AI, that philosophy is even more important in how we operate.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> The real pivot moment for me was when we fully started integrating security into DevSecOps. That was the first time I really understood this wasn't just a technical function, it was a company-wide shift. At that point, I realized I could actually be a blocker if I didn't evolve. I was suddenly being pulled into contracts, discussions, and conversations where people were talking about technologies like Docker and Kubernetes, and honestly, I had no idea what they were talking about at first.
>
> That experience made me step back and realize I needed to be more involved in what engineering and product were doing, not just thinking about security in isolation or purely from a backend perspective.
>
> From then on, I made a conscious decision to get much closer to engineering and product, and to start thinking in terms of business outcomes rather than just technical controls.

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> Networking across the organization and genuinely learning how the different parts of the business work.
>
> A lot of people stay within their own lane, especially in security, but you really have to get to know teams that don't seem immediately relevant: communications, HR, marketing, all of those groups. They have a huge amount of influence.
>
> Even small interactions with those teams can really raise your visibility and help you get traction in larger conversations. And in security specifically, that matters even more because you're already fighting an uphill battle. If you can build relationships across those groups, you end up with more trust, more influence, and people are more willing to bring things to you early instead of avoiding you.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> I don't even think it's five years out. I think within about three years the structure of security teams is going to look very different. I don't think I'll have security analysts anymore. My team will be security engineers who are managing agents, not just reviewing alerts. Their job will shift much more into understanding the business, working closely with engineering, and operating AI systems.
>
> The teams will also likely be smaller, and we'll probably see consolidation in the vendor space because more capability will live inside fewer platforms. If organizations get it wrong, it'll be because they didn't lean into AI early enough. They won't understand how to use it, they won't be managing agents, and they'll still be asking for headcount instead of showing how they're scaling through technology.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

> It definitely excites me. We're already seeing how much faster things are becoming. But it doesn't remove the human from the equation.
>
> Even when AI finds problems, there's still a lot of human interaction required, especially with engineering teams. That's why I think soft skills are becoming so important.
>
> How you work with people, how you communicate, how you explain risk… that's becoming a bigger part of the job than ever before. It's really about being able to see further and move faster, not replacing people entirely. The human layer is still essential.

AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

> Yes, I think it absolutely is table stakes now. We require our security engineers to be able to use agentic coding and AI tooling effectively. It's not optional anymore. They need to understand it not just so they can work faster, but so they can understand what engineering is doing and respond at the same speed.
>
> We're also pushing teams to build and run agents themselves. That's becoming part of the core skill set.
>
> At the same time, the expectations have gone up significantly. The scope they're expected to cover is much broader than it was even a year or two ago, because AI allows them to analyze more, faster, and across more systems.
>
> So yes, AI is absolutely being used on both sides. But for us, it's about increasing speed, increasing coverage, and making sure our teams can operate at the same pace as what's coming at them.

Jamie Dicken

Director, Security Platforms and Architecture

GitLab



Bio

Jamie Dicken is a pragmatic security leader who defines her career by the ability to bridge the historical divide between software engineering and security operations. Having begun her professional journey as a software engineer and engineering manager, she possesses a first-hand understanding of the frustrations developers face when security requirements lack business context or technical feasibility. This dual-perspective informs her leadership style, which prioritizes business alignment, clarity, and a "calm" approach to risk management.

Currently leading Security Platforms and Architecture at GitLab, Jamie is a vocal advocate for decentralizing security responsibilities. She challenges the traditional "anti-pattern" that security work belongs solely to security teams, arguing instead that software teams should own the security of their systems through self-service "golden paths" and automated guardrails. Jamie is currently focused on the strategic evolution of the Agentic SDLC, experimenting with in-house AI agents to perform code reviews and threat modeling to match the increasing velocity of modern R&D.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> For me, it was my own experience. I started as a software engineer and manager, and I frankly struggled working with security teams. It felt like they got in the way more than they helped, and their requirements often didn't make sense for what we were building. Later, when I moved into security, I had the opposite frustration: working with engineering teams who didn't always grasp the severity of certain issues.
>
> The experience on both sides of that table has been key to my passion for helping software teams and security engineers work better together toward the shared goal of building quality, reliable, and secure software. That requires both business alignment and pragmatism.

What is the best piece of advice you have ever been given that you still apply to your career today?

> A mentor told me I didn't have to figure out my entire career all at once. I couldn't have predicted my own path, even ten years ago. I wouldn't have expected to move into security. There's also a seasonality to life. At different points, your capacity and priorities change.
>
> Early on, taking on more responsibility wasn't realistic. Now, I have more time and energy to invest in my career. That will change again.
>
> The key takeaway is not to rule out future opportunities based on your current circumstances. Stay open to what's possible.

Jamie Dicken, Director, Security Platforms and Architecture

GitLab

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> The ability to communicate with clarity. If your message isn't concise, it won't land. And if you can't explain why something matters to the business, it won't get addressed.
>
> Calm is just as important. Not everything can be critical. If everything is urgent, nothing is. You have to calibrate your response so conversations stay productive, not combative.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> In the agentic AI age, we are facing an open source crisis. Many open source libraries already depended on single, unpaid maintainers. Now they're dealing with AI slot submissions and a deluge of AI-enabled vulnerability reports, which is causing them to either shut down bug bounty programs, archive projects, or simply go radio silent.
>
> If all of a sudden we can no longer rely on the open source libraries that power our platforms, it becomes a long-term compatibility, quality, and reliability risk. I see that as an existential threat to the value delivery of our businesses.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> The idea that security teams should be the ones doing all the security work. Threat modeling, triaging vulnerabilities, reviewing code...those can't sit solely with security.
>
> Software teams own the security of their systems, just like they own quality and functionality. As organizations scale and adopt AI, relying exclusively on security teams leads to worse outcomes.
>
> The role of security should be to build guardrails, or what you might call "golden paths", that make it easy for software teams to do the right thing by default.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> Getting AI right starts with not being afraid of it. You can't secure what you don't understand, and you can't scale manual AppSec processes to match AI-driven development speed.
>
> If we engage with it, we can define what a secure, AI-enabled development lifecycle looks like. If we don't, the gap between development speed and security coverage keeps growing.
> That leads to constant reactivity: firefighting instead of prevention.
> That's what getting it wrong looks like.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

> I'm generally optimistic. As an industry, a key reason why we have so much security debt today is that security considerations are generally made way too late, where it becomes prohibitively expensive to change course.
>
> The promise with Agentic AI is that code development can become cheap, or even free from the human cost perspective. You can fix or add functionality just as easily as you can use those same tools to address security concerns. We could actually pay down the security tech debt we currently have, and that holds a lot of promise.

AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

> We're now building our own agents to perform security-focused code reviews, create threat models, and identify or remediate vulnerabilities. Just as important, we're building a team of people who are really passionate about reimagining what software delivery looks like in this new era of Agentic AI.

Daniel Hammon

Director, Information Security and Compliance

Signifyd



Bio

Daniel Hammon brings a unique and high-energy perspective to cybersecurity leadership, rooted in a diverse career path that spans the culinary arts, finance, and academia. Formerly an executive chef with extensive experience in high-pressure, live-media environments, Daniel transitioned into security by way of GRC and auditing before diving deep into the technical "rubber meets the road" realities of offensive security. This rare blend of skills allows him to translate complex technical risks into clear, calm, and actionable business narratives.

Currently the Head of Information Security and Compliance at Signifyd, Daniel is a passionate advocate for building a security culture that empowers every employee to be a line of defense. When he isn't securing the product lifecycle at Signifyd, Daniel mentors the next generation of talent as a cybersecurity lecturer at California State University, Sacramento, where he emphasizes the critical importance of public speaking and research in the security field.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> I came up through security in the GRC audit world and then progressively got more technical. I did things like Security+ and CISSP, and I was feeling pretty confident, but I didn't truly understand security risk until I saw my first demonstration of a reverse shell.
>
> It was mind-blowing. As soon as I saw it, I set up a lab and executed it myself, and everything clicked. I said, "This is what it's really all about. This is where the rubber meets the road when it comes to security". That moment fundamentally changed how I think about security.

What is the best piece of advice you have ever been given that you still apply to your career today?

> It's a little corny, but chase your passions.
>
> I've had careers in the restaurant industry and in banking, but when I found security, I was so enamored with it because I'm motivated by learning. There's so much to explore (wide and deep) and you can push the knowledge of the industry forward. Chasing your passions makes you truly happy; it doesn't even feel like work.

✦ **Daniel Hammon,** Director, Information Security and Compliance

Signifyd

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> I often say that it's not folks like me that keep an organization safe. It is everybody throughout the organization making the right decision at the right time. The most important thing that we do is develop a security culture at an organization. When it's top of mind, people know what to do and how to do it. It builds confidence and enables the business. At Signifyd, probably because of its fraud background, security is intuitive to every department from our leadership all the way down to new hires.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> Meeting baseline security requirements, like SOC 2, isn't really the point. But if you put in that effort and "show your work" (like algebra homework) the audits fall in line.
>
> We end up far exceeding those minimum requirements. That makes audits a breeze and gives our customers confidence because we are enthusiastic about doing the right thing, the secure way.

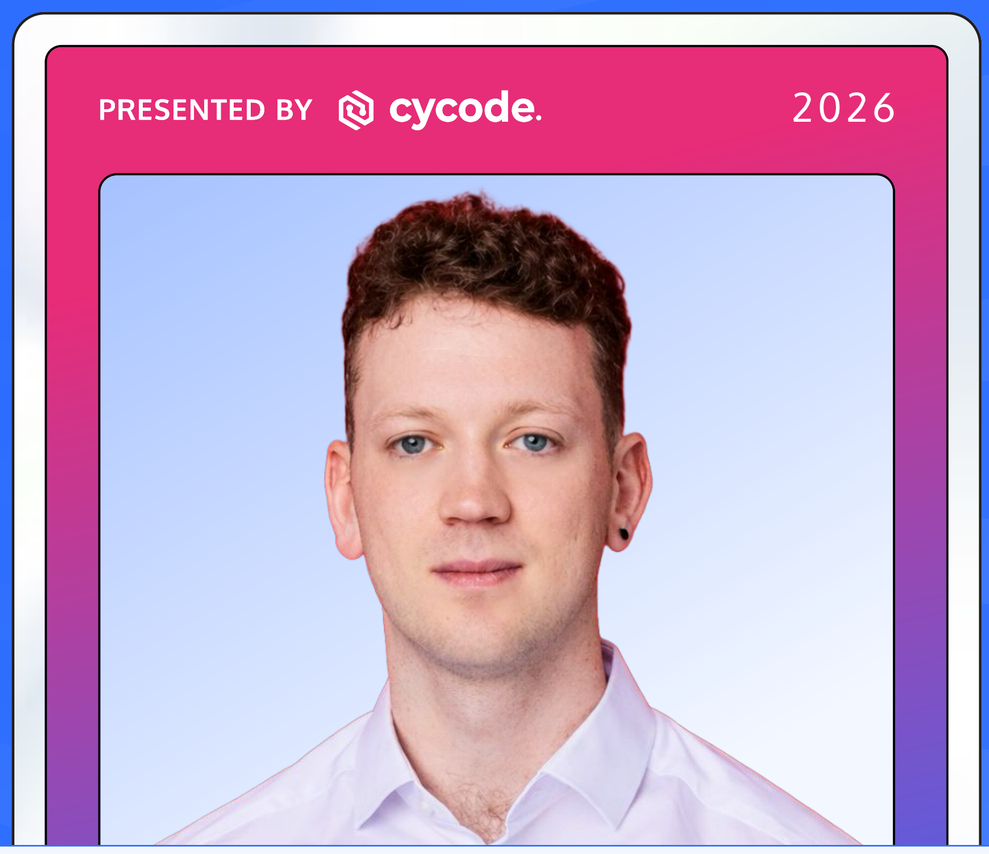
Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> Code is logical, and AI scales our ability to create it rapidly. We're moving toward what I call an AIDLC, the AI Development Lifecycle. PRs and commits are getting bigger as AI accelerates coding, so the lifecycle has to adjust. If we get it right, AI becomes part of the workflow, empowering teams while keeping quality high.
>
> It's about instilling confidence and creating guardrails, not just automation for its own sake. AI can help predict vulnerabilities, guide development, and support security in ways we've never had before, but human oversight, clarity, and transparency are critical to make it work.

Conleth Kennedy

Manager, IT Application Security

unum



Bio

Conleth Kennedy is an accomplished security leader and software engineer with over eight years of experience bridging the gap between deep technical engineering and organizational security strategy. Currently serving as an Application Security Manager, Conleth has a proven track record of leading DevSecOps initiatives, ranging from building CI/CD maturity and automated scanning suites (SAST, SCA, DAST) to managing complex microservices architectures.

With a background that spans full-stack development and AI-driven automation, he brings a unique "engineer-first" perspective to security. Conleth is a strong advocate for "strategic persistence" and cultural transformation, focusing on behavioral incentives and "policy as code" to ensure security acts as a business enabler rather than a friction point. He holds an Honours Bachelor of Science in Software Development from the Institute of Technology, Carlow.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> For me, it was an episode of the Application Security Podcast on OWASP SAMM.
>
> I'd always been very technical and focused on defects and controls. But that episode shifted my thinking toward the behavioral side of security. It looked at how organizations actually operate, and how that impacts engineering.
>
> It made me think more about incentives, how you measure maturity, and how teams make decisions. Not just what controls you have, but how people interact with them.
>
> That still holds up, because those underlying dynamics haven't changed. If anything, they've become more important.

What is the best piece of advice you have ever been given that you still apply to your career today?

> Don't stop at the first answer. Always try to get to the root cause. That's been a big part of how I approach problems. I try to really understand what's going on before jumping to a solution.
>
> Sometimes it's technical. Sometimes it's a process gap, or even cultural friction. If you stay at the surface level, you miss the real issue. So I try to go deeper first, then respond.

✦ Conleth Kennedy, Manager, IT Application Security

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> Strategic persistence. A lot of the time, it's not about knowing the right answer, it's about staying with the problem long enough to get there. You need a bit of grit to keep the conversation going and push for the right outcome, even when it's inconvenient.
>
> At the same time, you have to do it in a respectful way. It's about being consistent and steady, even when it feels uncomfortable.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> The democratization of software development.
>
> We're no longer just securing applications, we're securing AI-generated code, often created by people who aren't traditional engineers. Anyone with a laptop and a subscription can build something now. But not everyone understands how to secure it.
>
> That creates a kind of structural risk that we haven't fully acknowledged yet. In other fields (like aviation or engineering) you need qualifications. In software, you don't.
>
> It's a double-edged sword. It unlocks innovation, but it also increases risk at scale.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> We're going to have more software, and it's going to be more complex. That's a given.
>
> If we get it right, AI helps reduce technical debt, speeds up triage, and automates repetitive work. That lets people focus on more strategic, architectural decisions. Security becomes more proactive.
>
> If we get it wrong, we end up with unreviewed code, opaque logic paths, and AI-driven regressions. We'll spend more time reacting, rolling things back and dealing with the fallout.
>
> The difference won't come down to tooling. It comes down to the governance we put in place today.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> Heavy, manual policy gatekeeping.
>
> It slows down delivery, creates friction, and doesn't necessarily improve security. Instead, policies should be embedded as code, directly into pipelines and environments where developers are already working.
>
> That allows developers to be more self-serving through automation. You improve velocity, and security improves alongside it.

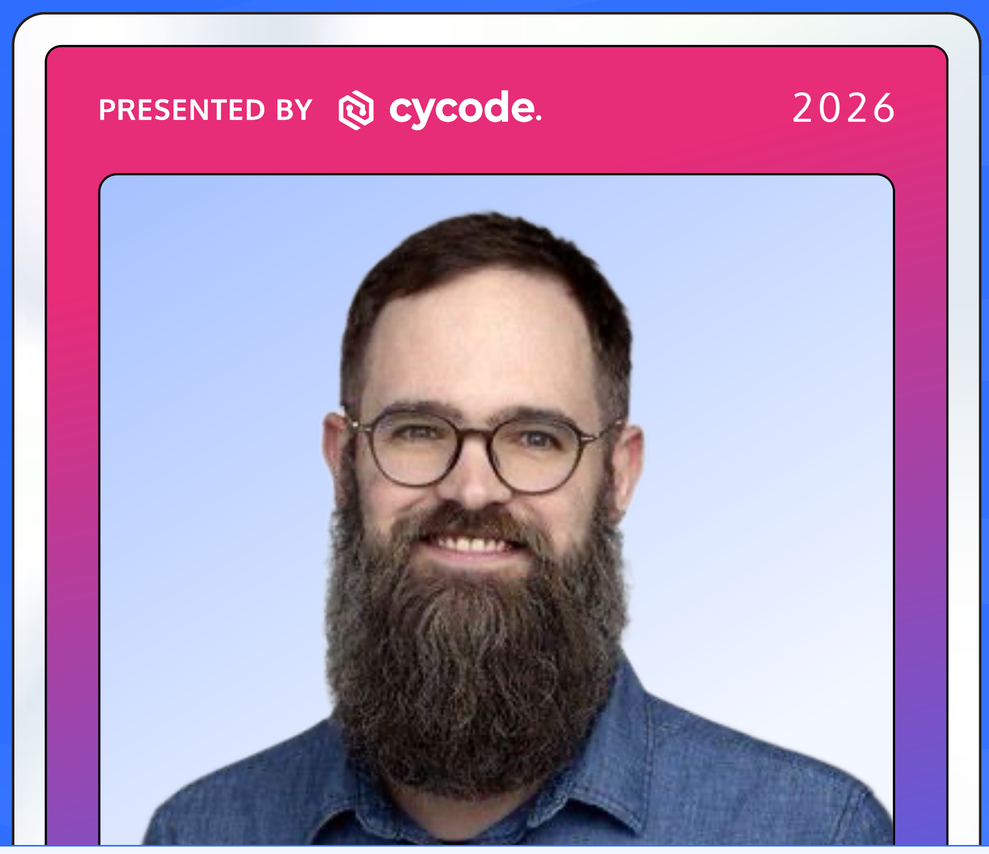
AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

> I think it's the price of admission. Attackers already operate faster than any human could possibly respond, so defenders have to evolve our detection and strategies. We have to figure out a way to identify patterns of agent-based attacks, and not just signatures or indicators.
>
> For me, the differentiator will be actually understanding AI on a fundamental level: how it behaves, how it fails, and how to react to those failure modes. That's where I'm investing in skills and program design today.

Kevin Markley

Senior Manager of Product Security

VEEAM



Bio

Kevin Markley is a Senior Product Security Manager at Veeam, a global leader in data resilience. With over 15 years of experience in cybersecurity and systems engineering—including a tenure at Veritas—Kevin specializes in scaling security within high-velocity, complex enterprise environments.

Throughout his career, Kevin has focused on transitioning security teams from traditional "gatekeeper" roles to a model driven by automated governance and "paved paths." He is a vocal advocate for context-first security, believing that the future of the industry lies in empowering developers to move at machine speed while security engineers act as strategic orchestrators.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up?

> Everything I do in security kind of comes from everything else I've done in my career, but there was one moment early on that has really stuck with me.
>
> I was leaving the help desk and trying to break into DevOps for the first time, and I went into an interview thinking I had completely bombed it. They were asking me things like, "Do you code?" No. "Have you worked with AWS?" No. "Do you understand this?" No. And I basically just kept saying no to everything. I remember thinking, "Alright, I'm never going to hear from these guys again."
>
> But I ended up getting the job anyway, and it was really because of my background. I used to be a restaurant manager, working these 12-hour shifts, and then staying up late teaching myself IT so I could get out of that work. And the hiring manager told me something I've never forgotten. He basically said, "I can teach anyone to code, I can teach anyone the technology. What I can't teach is how not to be a jerk, and how to actually work hard to learn things."

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> I always tell my team this: we have to be like Ted Lasso's goldfish. We have to be agile, and willing to pick things up and drop things instantly. I might ask you to work on something for two weeks, and then the org changes, and we never touch it again. And you just have to be okay with that.
>
> Technology is evolving so fast, and even things from a year ago (especially with AI) are completely different now. So the key skill is adaptability, and just the willingness to constantly challenge what you think you know. Even something you thought was the right way six months ago might not be the right way anymore.
>
> So it's always about asking, "Is there a better way to do this?" and not getting stuck in old patterns just because that's how we used to do it.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

One thing I think is becoming an anti-pattern is severity-based prioritization done in a very rigid way. Like, you see a CVSS 9.9 and everything stops. Engineers drop everything, sprint cycles get broken, and the whole system gets disrupted without real context. That doesn't really work anymore.

What I've started doing instead is using AI models to contextualize risk. I'll take findings and send them to two different AI models, along with our architecture and system context. I'll ask them: what's the real risk here? What are the mitigating factors? How urgent is this?

I even make them argue with each other. And they come back and usually agree on the top risks. We've seen false positives drop by something like 95%. Then a human still steps in, adds context, and makes the final call.

Still, what used to take hours of manual analysis is now done in minutes. It's not about trusting one model, it's about cross-checking multiple models and then applying human judgment at the end.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

If AppSec teams get AI wrong, it's basically because they rely on it too much without understanding it, or they treat it as a replacement instead of a tool. That's dangerous, because attackers are absolutely going to use AI to inject supply chain risk and exploit how much trust we place in models.

We're already heading into a world where most code is AI-generated. So the real shift isn't syntax anymore, it's understanding architecture, patterns, and hallucinations. A win looks like security teams evolving how they work. Not focusing on formatting or line-by-line reviews, but focusing on whether the architecture is secure and whether the AI-generated code actually matches intent.

Like the shift from card catalogs to Google, we don't need to know every detail of Python anymore, but we do need to understand outcomes and be able to verify what the AI produces. If we get it right, security engineers spend less time on tedious review work and more time building systems, guardrails, and automation that continuously improve security posture. If we get it wrong, we just blindly trust AI outputs and inherit risk at massive scale.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

It's really exciting, honestly, because the part of my job I don't like is reviewing massive amounts of code. I don't enjoy going through 3,000 lines trying to figure out what's going on. What AI does is remove the tedious part. It highlights what actually matters and says, "this section looks risky, go look here, everything else is boilerplate." And that's huge.

This has actually allowed me to start coding more in the last month than I probably have in years, even though I'm not a developer. Now I can actually build things I never would have been able to build before.

That's why I tell my teams "this doesn't replace you". It just removes the tedious work so we can focus on higher-value problems. Instead of reviewing repetitive PRs or configs, we can focus on building better risk systems, automation, and security tooling we never had time for before. So people are being judged more on ideas, not just output. And that's a good thing.

AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

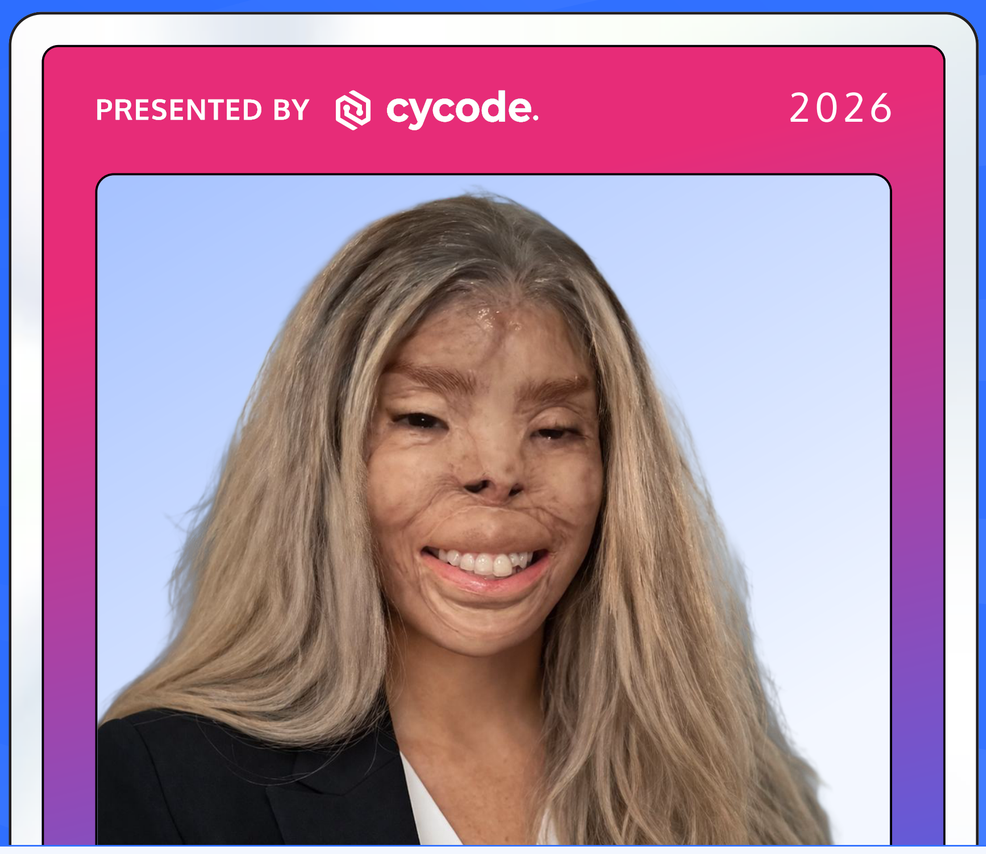
Yeah, it's definitely the price of admission now. If you're not using AI, you're going to get left behind. It's the same shift we saw with cloud adoption. We have to use AI to match the speed of attackers. That means threat hunting, detection, response...all of it has to move from hours to minutes.

We need agents that can run those hunts automatically, pull in threat intelligence, correlate findings, and surface issues without a human waiting on it. But even with that, humans still matter. We still need guardrails, and we still need judgment. The goal is speed, but with control layered on top.

That's really what it comes down to: AI-driven speed, with human-defined boundaries.

Kimberly Mattheys

Product Security Leader



Bio

Kimberly Mattheys is a dedicated security practitioner and consultant who champions a pragmatic, results-oriented approach to protecting the modern software pipeline. She is a strong advocate for "Secure Curiosity," a philosophy that encourages engineering teams to explore and break new technologies like AI, provided they maintain the human context and guardrails necessary to prevent decision manipulation.

Throughout her tenure at organizations like Solaris, Kimberly has distinguished herself by her ability to connect disparate technical dots into clear, outcome-based business stories. She is a vocal critic of the traditional "Shift Left" anti-pattern when it is executed without developer buy-in, instead advocating for "shifting where it matters" to ensure security accelerates rather than hinders delivery. By focusing on exploitability and developer enablement, Kimberly builds security cultures rooted in trust, transparency, and actionable intelligence.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> Not every vulnerability matters, only the ones that can actually be exploited. Early in my career, like most teams, we were measuring success by volume: the number of findings, coverage, the signals coming in from all our tools. But that didn't really translate to risk reduction.
>
> I shifted my perspective to focus on what attackers could actually use against us. Attackers don't care about theoretical issues, they care about viable paths to impact. That's the approach I still apply today: looking at what's exploitable cuts through the noise and keeps teams focused on what truly matters.

What is the best piece of advice you have ever been given that you still apply to your career today?

> One of my mentors, who had been my manager at Solaris, noticed something about me: I was excellent at explaining what was wrong and even providing workarounds, but that alone doesn't move security forward. He advised me to speak in outcomes, not just problems.
>
> I had to translate security risks into things the business actually cares about: regulatory impact, customer trust, revenue, uptime, and reputational risk. That's when I saw conversations start moving, when I stopped asking for permission and started shaping decisions. It completely changed how I approach my role.

Kimberly Mattheys, Product Security Leader

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> The ability to connect the dots and clearly tell the story. Security teams receive signals from various tools, incidents, and security champions, but individually, these signals aren't very helpful. The person who can synthesize everything, prioritize, and answer, "This is what matters, this is why, and this is what we should do about it" is the one who becomes a leader.
> That ability enables teams to act confidently and ensures security aligns with business objectives.

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> Early in my career, vulnerability scanning across our pipeline taught me a critical lesson. On paper, we had visibility end-to-end and were following all the maturity checklists, but developers had lost trust. Alerts were overwhelming, many weren't actionable, and the findings weren't being addressed. We had to rethink our approach.
>
> If we weren't reducing risk, we were just manufacturing 'checkbox security.' We used filtering like reachability and exposure to drill down into what was true and exploitable. The alerts were fewer, and teams started to trust us again because the violations were actually affecting production.
>
> It's about risk; if you're not reducing risk, you're not actually doing security.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> Decision manipulation. Most people focus on supply chain risks or AI directly, but attackers are increasingly manipulating business logic and influencing outcomes through indirect inputs. I saw an example where a customer tricked a chatbot into generating a 75% discount and then threatened legal action if it wasn't honored.
>
> It highlighted that AI and automation aren't inherently safe, they need guardrails. Teams need to think beyond obvious vulnerabilities and anticipate how models can be misused to impact the business.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> Shift left, for sure. The intention is right, but if it's not executed properly, it creates a lot of noise and friction with developers. I'm very big on developer enablement. Shift left can become noisy because my role is inherently invasive: everything I do impacts product, operations, and security. I need to bring people along on the journey, not just expect them to fall in line.
> With AI, code is being generated far too quickly for traditional human code reviews anyway. What works better is to enforce where it matters. Give developers context early, but don't block them unnecessarily. Put strong controls at high-impact points like release gates and runtime, and focus on what is actually exploitable to cut through the noise.
> That's how you build buy-in and trust, because the signals are real, and you're not slowing down delivery.

Cássio Pereira

Senior Application Security Engineer

StoneX



Bio

Cássio Pereira is a veteran security leader and software developer who spent over a decade in engineering before transitioning to cybersecurity. Currently serving as a Senior Application Security Evangelist at StoneX Group, a Fortune 100 financial services firm, Cássio specializes in bridging the gap between developers and security teams.

Cássio's strategy centers on four key pillars: Culture, Process, Tools, and Automation. He believes that in the age of AI, success isn't defined by tool volume, but by the human judgment required to govern them.

What is the best piece of advice you have ever been given that you still apply to your career today?

> It's a cliché, but do security from the beginning. I was a software developer for 12 years before switching to cybersecurity, and I still see companies making the same mistake. They build, release, and go to market, which is fine, but there are consequences. There's always a price to pay, usually in the form of incidents because details were overlooked.
>
> So my advice is always the same: start thinking about security early. You don't need to be overly restrictive, but you do need to pay attention to it from the beginning.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> We honestly don't know where this is going. The speed is incredible, but quality is the tricky part. I can cook very fast, but if I'm not a chef. That is to say it's not necessarily good... it's just fast. Software development is the same. It might work, but from a security perspective, fast doesn't mean good.
>
> With experience, I can guide AI to build more securely, using prompts that consider threats. But not every developer has that background. That's where the risk is. The tools are powerful, but the outcomes depend heavily on who is using them and how.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> Blind scanning. Everyone knows they need SAST or DAST tools, but many people use them blindly. They trust the scanner because they aren't experts in security.If you don't know how to configure the tool properly, or how to ensure your code has the right coverage, you end up trusting a report that says you're secure when you might not be. Scanners are just one tool. Blind trust in them creates noise, confusion, and false confidence.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

> Both. It excites me because it can prove that a vulnerability was fixed quickly. But it terrifies me when the person using it doesn't understand how a feature could be abused. AI in inexperienced hands can lead to chaos. It's like drinking and driving. You can't stop people from doing it, but if something goes wrong, the responsibility is still there.

AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

> We focus on four pillars: culture, process, tools, and automation. Everyone in the company needs to understand why security matters. If one of those pillars is missing, AppSec breaks down, whether you're writing code manually or using thousands of AI agents.
>
> We also focus heavily on resilience. We don't know how to fight the next zero-day any more than we knew how to handle something like COVID when it first appeared. There will always be unknowns. So the goal is to be ready, to absorb the impact, recover quickly, and minimize damage. Attackers have no limits, so we can't afford to limit how we defend ourselves.

Rusty Perry

Enterprise CISO Leader



Bio

Rusty Perry is a seasoned cybersecurity executive with a career defined by leading global strategy and architecture for organizations ranging from agile startups to Fortune 150 enterprises, including leadership roles at Western Union, EchoStar, and DaVita. A former Army veteran, Rusty brings a disciplined "impact over activity" philosophy to security, focusing on shifting the narrative from restrictive control to "safe velocity".

He is a staunch advocate for "influence without authority," believing that true security success stems from building political credibility and aligning technical risk with business outcomes. Currently deepening his expertise through the Stanford Graduate School of Business LEAD Program, Rusty specializes in architecting the governance required to secure the next generation of digital platforms.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> Security is not really about control; it's about enabling safe velocity. Early in my career, security was very control-oriented (blocking things and checking boxes) but at scale, that breaks down. The business doesn't slow down; it routes around friction.
>
> Security only works when it aligns with how the business is building. It's not just about reducing findings, but about enabling teams to move faster with better visibility.

What is the best piece of advice you have ever been given that you still apply to your career today?

> Don't confuse activity with impact. We are great at generating activity (scans, findings, reports) but none of it matters if it doesn't change actual risk or business outcomes.
>
> From my military background, another one stuck with me: you earn the right to be heard before you earn the right to lead change. You have to build bridges and collaborate, or your changes will just be quietly ignored. It's about building a partnership so the business actually wants to work with you.

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> Influence without authority. In product security, you don't own the teams building the product. Success comes down to credibility, clarity, and alignment, not control. You have to translate risk into a business decision, not just a technical report.

✦ **Rusty Perry,** Enterprise CISO Leader

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> In a recent API security review for a global company, what looked like a standard review surfaced an existential revenue risk. By aligning security, fraud, and customer experience data, we were able to reduce fraud exposure while significantly improving customer satisfaction and decreasing friction. It was a clear example that security can be an enabler, not just a point of friction.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> If we get it right, security is embedded into how software is built (security by design) with real-time risk visibility and automated remediation. AppSec teams may shrink in headcount but grow in influence by leveraging their own agents. If we get it wrong, we're just scaling insecurity faster. The difference comes down to governance. We need to build the governance architecture before the risk accelerates past us.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> Identity fragmentation, especially with Agentic AI. We've spent years focusing on code and APIs, but we aren't talking enough about authentication and authorization for AI agents. We're moving from securing human-driven systems to securing machine-driven decisions, and most security programs aren't organized for that shift yet.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> Treating vulnerability counts as a measure of security. It's just whack-a-mole, and in an AI-driven world, it completely breaks because the volume of noise is too high. What matters is exposure: what is actually exploitable and running at runtime. We need to shift from vulnerability management to exposure management tied to business impact.

Chase Pettet

Principal Security Engineer

Life360



Bio

Chase Pettet is a veteran security leader and architect with over 20 years of experience building secure, scalable software across highly regulated industries. Currently a Principal Security Engineer at Life360, Chase is a specialist in driving product security automation and risk-informed decision-making. His career spans critical roles at the Wikimedia Foundation, Illumina, and Archer, where he founded the Product & Application Security team and scaled security programs to support massive SaaS growth.

A builder by nature with more than a decade of hands-on coding experience, Chase excels at turning complex compliance requirements, including HIPAA, PCI DSS, and FedRAMP, into practical, value-preserving engineering solutions. He is a vocal advocate for collaborative security cultures and the evolution of cloud-native DevSecOps.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> I don't have a singular book or podcast. What changed how I think is more of a mental exercise I built over time. I call it the "root-end principle," inspired by something oddly specific: bagged salad. You always get those inedible root ends mixed in. And I found myself thinking, what would it actually take to remove those entirely? New machinery, oversight, process changes... real cost.
>
> That became a way of thinking about product decisions. Not everything imperfect is worth fixing. Every control has a cost, and every decision reflects what an organization is willing to tolerate.
>
> So in security conversations, I ask: is this a real risk to users, or just something that offends my professional sensibilities? Only one of those matters. It's a way to stay grounded and focused on actual harm, not theoretical perfection.

What is the best piece of advice you have ever been given that you still apply to your career today?

> The best advice I've ever received was that the only way to succeed is to succeed sustainably. Security work is endless. There's always more to fix, more to investigate, more to improve. If you approach it by constantly grinding, you burn out, and worse, you become ineffective.
>
> That really shows up in incident response. The instinct is to swarm the problem, but if an incident runs long enough, exhausted responders become a liability. The right move is often to rotate people out, not pile more in.
>
> An exhausted defender is an attacker's best friend, so you have to manage your energy like a finite resource. Progress isn't about intensity, it's about direction and consistency over time.

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

Integrity. You can't fake it, and once it's gone, you don't get it back. It's what enables everything else, particularly humility and transparency.

In security, you're constantly part of risk trade-off decisions. People need to believe that you're being honest, that you don't have a hidden agenda, and that you're not trying to block progress for its own sake.

I often say: security's job isn't to stop you from jumping into the gorilla enclosure, it's to make sure you understand that you're doing it. That only works if people trust you. And trust starts with integrity.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

The biggest under-discussed risk isn't just AI itself; it's what happens when we start outsourcing judgment to it. AI systems are built to converge toward the mean, and they're excellent at producing "average" answers. But security (and decision-making in general) often depends on recognizing edge cases and applying context.

What worries me is the slow drift toward treating AI as a final decision-maker. Not just for implementation details, but for strategy and judgment. You can imagine subtle, persistent manipulations, like a long-term prompt injection that biases decisions in small ways. Would anyone notice? Maybe not.

We've seen this pattern before. Early concerns about Wikipedia weren't really about accuracy, they were about people replacing judgment with convenience. AI risks taking that even further.
If we adopt its limitations as our own, eventually that catches up with us.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

Static least privilege, as we often implement it, is mostly baloney. In theory, it makes sense. In practice, it's vague and constantly shifting. What does "least" actually mean? The permissions someone needs in five minutes are different from what they need in a month. It is like the loitering and jaywalking of the security world. We pull it out when it's convenient, but it has very little fixed meaning in a practical business context.

What we end up with is a compromise that feels safe but isn't clearly defined or consistently enforced. More importantly, it can actively work against what organizations say they want: innovation, collaboration, and discovery. Those things require context and exposure, not restriction to the bare minimum.

This creates a huge disadvantage in the era of AI and innovation. Discovery and collaboration cannot happen with the bare minimum of rights and context. It's pretty untenable. If you want the developer, or AI agent, to put the pieces together and come up with a working solution, you need to expose them to more than the minimum information.

I'm not arguing for "maximum privilege." But we need to be honest: what we call least privilege today is often a loose proxy, not a precise control, and sometimes it delivers more friction than value.

Agentic AI can now autonomously write code, find vulnerabilities, and ship fixes without a human in the loop. Does that excite you, and what does it mean for the humans on your team?

It's both exciting and terrifying.. but mostly exciting. This is another foundational shift. The scale of output changes, but the core problem doesn't. We still have a software factory, and it still needs assurance.

What does change is who participates. I've started using the term "software creators" instead of developers, because more people (across finance, operations, everywhere) are now generating code.

That means responsibility expands, too. If you're creating software, you're accountable for its outcomes, whether you wrote every line yourself or not.

So the future isn't about removing humans, but about redefining their role, and making sure the systems we build can operate safely at a scale we can no longer manually control.

Michael Westphal

DevSecOps Lead

HARMAN
A SAMSUNG COMPANY



Bio

Michael Westphal is a veteran software architect and DevSecOps leader who specializes in the rigorous security requirements of the premium automotive market. With over a decade of experience at HARMAN International, Michael leads the design and rollout of modern DevSecOps processes, overseeing a global technical footprint that spans microservices, Docker-based CI/CD, and the complex Java-based NDS.Classic database compilers as well as Android and C/C++ applications that power high-end navigation systems.

Michael's approach to security is defined by his belief that staying ahead of the threat landscape requires a relentless drive to understand what is possible today while preparing for what might come next year.

What's the one book, podcast, or idea that fundamentally changed how you think about security, and why does it still hold up today?

> When I was first introduced to security, I wasn't fully aware of everything required to truly secure an environment, especially what can happen during development.
>
> Seeing security issues within the Cycode platform (even in inactive code) made me realize how easily sensitive knowledge can reach the wrong people. It taught me that it is critical to respect these risks and have simple, effective ways to detect them.

In your view, what is the single most important non-technical skill an "All-Star" must possess to influence a modern organization?

> Communication is the most important soft skill. You have to be able to reach out to people, share information, and never be shy about asking questions.
>
> It is vital to understand the perspective of others (especially when you are a technical person talking to non-technical stakeholders) and translate complex topics so everyone understands exactly what you are talking about.

What is the best piece of advice you have ever been given that you still apply to your career today?

> Quite simply: don't stop learning. Be curious. You have to stay up-to-date and be prepared for anything to happen.

Michael Westphal, DevSecOps Lead

HARMAN
A SAMSUNG COMPANY

What is a specific project or "calculated failure" that made you realize Product Security was a strategic business driver rather than just a technical hurdle?

> Online services are a prime example. If your platform is vulnerable, the problems quickly leak into your internal systems. Once bad actors find a way in, you've already lost. This highlights why it's a business necessity to stay current and ensure there are no open gaps.
>
> Even small misconfigurations or overlooked issues can have outsized consequences on business operations and customer trust, and seeing that play out repeatedly makes it clear that security is integral to the company's ability to operate effectively.

What is the "silent" threat at the product layer that isn't getting enough boardroom attention yet?

> While everyone is focused on the latest technology and new products, we have legacy products that have been maintained for years. You have to deliver patches and updates, but you aren't always able to update those systems to the latest versions. Ensuring that outdated infrastructure remains as secure as possible is a very critical, overlooked topic.

Five years from now, what does a world where Application Security teams got AI right actually look like, and what does it look like if they got it wrong?

> If used correctly, AI will help prevent risk by providing forecasts and securing systems in advance before an issue even occurs. I hope to see it evolving to perform tests and consider scenarios that humans might not even think about.
>
> If it goes wrong, we will be attacked by incredibly clever AI-based agents. Companies will suffer from these attacks and lose business secrets or sensitive data they never intended to share. You must be prepared for the fact that if you are using AI, the attackers are using it too.

What is one widely accepted Product Security "best practice" that you believe is actually an anti-pattern in the age of AI, and what should replace it?

> For years, it was accepted that simply scanning your existing code and declaring it "secure" was enough. But static scanning doesn't consider what is possible with AI. For example, people often accidentally leave company secrets in private repositories that later become public. Previously, no one thought to scan the personal repositories of former employees, but we've seen that this is a real, active risk that AI-powered solutions can now detect.

AI has scaled attacker capabilities to a level no human can match. Since 'fighting AI with AI' is now table stakes rather than a competitive edge, how are you evolving your team and strategy to keep pace?

> We are training our developers to use AI and encouraging them to explore what is possible, but we are also making sure they understand the consequences and side effects. It's necessary to make people aware of what is possible today and what we imagine will be possible a year from now to give them the right perspective.

Closing the Security Gap

Security teams—like every other function—now have to adopt AI to keep up. But they face a unique challenge: the same technology they're using to scale is also accelerating the threats they need to defend against.
What stands out across these conversations isn't just what's changing, but how leading teams are responding to it.
The Product Security All-Stars aren't adding more tools or tightening controls. They're rethinking the model entirely.
Security is no longer about blocking progress or chasing every vulnerability. It's about enabling teams to move quickly, with clarity on what actually matters.

That shift starts with context.

In a world where vulnerability volume is infinite and AI accelerates both development and exploitation, raw signals are no longer useful on their own.

What matters is understanding which risks are real, which paths are exploitable, and what actually impacts the business. This requires a Context Intelligence Graph that offers a unified view across your software factory from code to runtime. This is the only way teams can prioritize with precision instead of reacting to noise.
But context is just part of the equation. We can't ignore the fact that the role of the security team is fundamentally changing.

In a race to drive efficiency at scale, AI is now handling much of the repetitive work: generating code, identifying issues, and even remediating them. The differentiator is no longer how much work a team can do manually, but how effectively it can guide, validate, and govern these systems. In response, security leaders are shifting from operators to decision-makers, focusing on judgment, context, and ensuring the system is working as intended.

The organizations getting this right are converging on a similar winning formula:
Culture: Security is a shared responsibility, not a centralized function
Process: Controls are embedded where they matter, not enforced everywhere
Context: Prioritization is driven by real, exploitable risk, not volume
Autonomy: Remediation and response are automated to match the speed of development

This is what it takes to operate in the age of AI. Shift to AI, securely with Cycode.



Secure AI adoption without slowing development

Cycode helps security teams discover AI across the SDLC, govern what's allowed, protect developer workflows in real time, unify AI risk, and orchestrate action across the software factory.

Get a Demo →