SECURITY JOURNEY

White Paper

Education vs. Awareness

Why a Programmatic Approach to Secure Coding Training Matters.

Education vs. Awareness: Why a Programmatic Approach is Better

This paper is the distillation of thoughts and ideas shared during the 'Security Journey Education vs. Awareness Roundtable' that took place in September of 2022. Thank you to the following thought leaders for providing their experiences and insights for us:

Jen Fox
Program Manager of Security Awareness and Education at Datadog

Jason Hong
Professor in the Human Computer Interaction Institute at Carnegie Mellon University School of Computer Science

Joe Ferrara
CEO at Security Journey

John Campbell
Director of Content Engineering at Security Journey

Amy Baker
Security Education Evangelist at Security Journey (Moderator)

Developers and everyone working within the software development lifecycle (SDLC) are under pressure to deliver 100x the volume of code than they were ten years ago. And while they're expected to create scalable, secure, and successful applications, they often don't have the education or knowledge for all three. Many developers do not have a computer science degree, and those that do will not likely have been taught one of the most critical elements of application development: ***security.***

According to Forrester, none of the top 50 undergraduate computer science programs in the U.S. requires a course in code or application security for majors.[1] This is especially concerning in the current era of serious application risk from sophisticated and evolving hacking techniques. In fact, the number of new vulnerabilities within the NIST National Vulnerability Database increased by over 200% from 2015 to 2021.[2] This lack of security education is unsustainable.

This is not to say that industry is unaware of the issue of insecure coding. Open source and software supply chain vulnerabilities are often making headlines – with industry bodies like NSA (National Security Agency) offering guidance,[3] and the U.S. government driving security requirements.[4] And each October, CISA (Cybersecurity and Infrastructure Security Agency) and NCA (National Cybersecurity Alliance) lead a collaborative campaign to raise cybersecurity awareness both on a national and international level.

Many acknowledge the problem – but this is no longer enough. We need to go further and embrace *education*, not simply awareness, around secure coding practices.

Moving Past Awareness to Education

We took the issue to a roundtable discussion; asking experts from both industry and academia:

- Where does awareness stop and education start?
- How is secure coding education made possible?
- What can we do to start bridging the gap between academia and private industry to ensure safe application development is a consideration from the beginning?

One central driver behind the Security Journey Roundtable was to find out how industry and academia are moving past 'awareness' and towards 'education'.

1: IBM, 2: Bugcrowd, 3: Sourcegraph, 4:Forrester

In general, security 'awareness' in application development = ***recognizing*** what a particular flaw may look like.

But security 'education' = ***understanding exactly how*** this flaw will affect the product, the business, and the customer and what can be done to remediate the flaw.

Where 'awareness' is knowing of a problem, 'education' is knowing what to do about that problem. While the two areas may be complementary, there is a big gap between 'being aware' of an issue and recognizing when it's in your code with an understanding how to fix it.

Unlike awareness, education in secure coding is founded upon the 'big ideas' of security. Not every situation and scenario can be taught, but with continuous and programmatic education initiatives, those in the SDLC can learn to recall issues that happen in new environments and begin to recognize patterns.

This secure coding education can often lead to 'code smell'. E.g., once patterns and principles are learned, teams will be able to identify code that isn't quite right, or notice a situation that creates risk – like un-sanitized data input – and design a solution to mitigate this risk.

But security **'education' = understanding** exactly how this flaw will affect the **product**, the **business**, and the **customer** and what can be done to remediate the flaw.

The Non-malicious Insider Threat

The onus for low-security knowledge shouldn't be placed on the individual developer – or any one role in the SDLC. There are too few education opportunities and resources available and the complex threat environment is increasingly difficult to navigate.

Just recently, we've seen large-scale data breaches on global organizations like Twitter, Uber, and Mailchimp, all initiated with social engineering attacks. These incidents are perfect examples of how vulnerable we all are when up against cybercriminals; anyone can fall victim to social engineering.

In the roundtable, we posed the question of whether, given the increased threat surface and lack of secure coding education, the developer could ever be considered a 'non-malicious insider threat'? The term 'insider threat' can be interpreted in a number of ways, but here we specifically mean those with no bad intent who unknowingly allow weaker security practices.

Unfortunately, anyone within an organization – from developers to HR managers to sales and marketing teams – can pose a threat in today's cyber landscape. Yet one key difference is the volume of important decisions that developers have to make in their day-to-day, often under significant pressure.

Developers therefore have an opportunity to **pose an even greater threat**, and **serve as an even better source of protection**, than other members of the team.

Developers therefore have an opportunity to pose an even greater threat, and serve as an even better source of protection, than other members of the team. To avoid accidentally becoming a non-malicious insider threat, developers should:

- Recognize the impact they have on their organization's attack surface
- Possess the knowledge to reduce these risks

Making Education Possible

Continuous education that goes beyond simply 'raising awareness' is so important to making sure developers avoid becoming this non-malicious insider threat. It needs to focus on 'shifting left' – creating a security-first mindset that ensures secure code is a consideration as early as possible in the development process.

This programmatic education starts with building foundations for coding securely, e.g., issues like hashing and buffer overflow prevention. But it also needs to address the gaps in knowledge and the most common mistakes that developers make. The technical and human aspects must be equally balanced from the start. It must become muscle memory for developers, and everyone involved in the SDLC.

For enterprises, investment in education needs to be driven down from the top. Key decision-maker support means no roadblocks to hinder progress and a much easier time shifting left. In fact, **the board should be leading by example and making sure everyone across the SDLC is thinking about security consciously – even if they do not have, or need, the hands-on knowledge for writing code.** This applies to anyone from Product and Program Managers to testers or User Experience (UX) Designers.

This education needs to be effective in order to drive adoption, which means it's:

- Relevant for each of these roles
- Bespoke to their experience
- Addresses the kinds of issues they face in their day-to-day

In fact, **the board should be leading by example and making sure everyone across the SDLC is thinking about security consciously – even if they do not have, or need, the hands-on knowledge for writing code.**

Support From Academia

The shift left initiative also needs support from areas like academia. Computer science professors should be examining their curriculum to make it easier for industry to embrace security from the start. At the moment, many introductory courses are focused on just correctness, efficiency, and performance. But security also needs to become a priority, to teach students:

- The value of coding securely early on
- How to spot basic security issues

Both industry and academia, must also consider stage models – e.g., the individual's 'stage' of knowledge and understanding, as well as where they are in their career. Education is never 'one size fits all' and context is so important to getting it right. If a team is simply labored with a training course when they're initially on-boarded, or while in the middle of other time-sensitive projects, it will be far less effective.

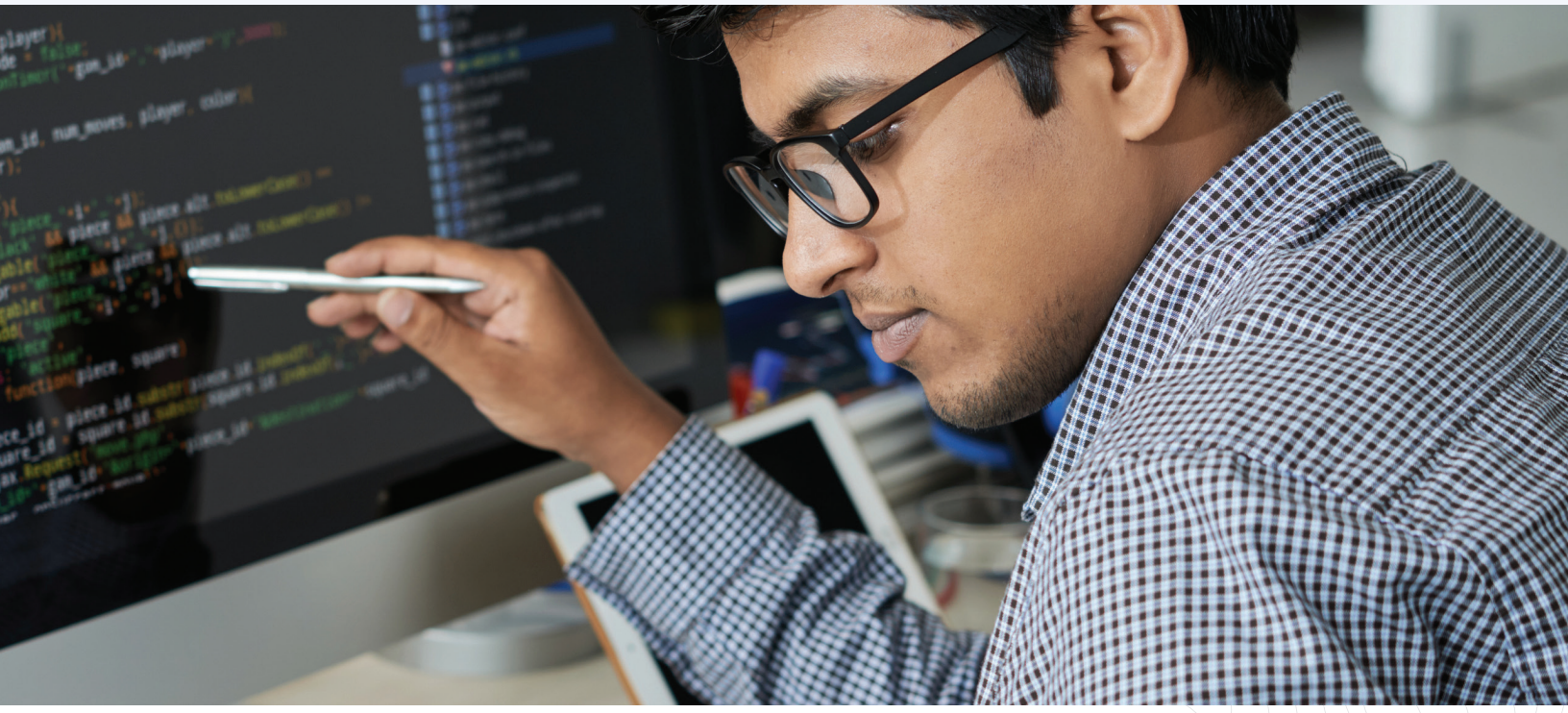
Regulation is Moving Forward

One day we'll reach a point where security is automatically baked in from the start and not considered an 'extra step'. But until then, we need to know how to drive a security-first mindset – for the SDLC, as well as organizations and industry as whole. This is where standards, regulations, and requirements come in.

For far too long, **it's been the wild west without clear and actionable advice or guidance around how to secure the software supply chain and improve cyber posture.** Software development is different from industries like transportation, which has a safety board to examine why and how things fail. And unlike manufacturing, it does not place safety as its number one priority, measuring and celebrating each day without an incident.

Therefore, the latest announcements from U.S. government on the requirements for their software supply chain, and guidance from OpenSSF on issues like SBOMs (Software Bills of Materials) is a positive step in the right direction.

For far too long, **it's been the wild west without clear and actionable advice or guidance around how to secure the software supply chain and improve cyber posture.**



But it Can Go Further

Yet there are still a number of issues with regulation and the industry needs to do more:

The U.S. government relying on 'self-attestation' for software supply chain security is not enough, and has led to many calling for more data-based evidence to prove compliance.

A lot of guidance lacks specificity: we need the development community to come together and codify exactly what some of these guidelines should look like in practice.

The industry is not learning from its mistakes. We don't analyze what went wrong and share key learnings to avoid it happening again. We need to measure outcomes and interrogate whether improvements have been made, instead of just ticking off checkboxes and complying with the bare minimum.

And for those that do fail to tick those boxes/neglect compliance, they should be more severely punished to incentivize others to manage their risk.

Finally, **although government action on software supply chain security is positive, it's now up to the businesses and enterprise decision-makers to rise to the occasion and take action too.** It's time for greater collaboration, with organizations working together in the public and private sectors to solve the issue of insecure code.

Shifting left = bridging the gap between academia and industry

Academia and industry both have a significant role to play in addressing the lack of secure coding education provided for the SDLC. For instance, CISA's 2023 – 2025 strategic plan5 focuses on how we can reduce risk and build resilience, drawing attention to the value of collaboration and information sharing, and how securing IT infrastructure is a joint responsibility.

The question is, how do we bridge the gap between what the industry needs their development teams to know, and what academia is providing at least for junior developers, when they're first coming out of college and coding programs?

We identified three key areas that will support this shift:

- Creating more platforms for sharing knowledge
- Encouraging more practical partnerships for organizations to solve issues together
- Embracing cross-pollination of knowledge through mentorship

Academics often publish research on privacy, security, new tools, and the human risk factor within journals and they attend scientific conferences. But they don't join industry at large security events like RSA. At the same time, industry rarely seeks advice or reads the research of academics in the same field.

This creates a serious divide, and there is clearly a need for more opportunities – venues, events, and platforms – to collaborate and learn from each other's research. **Professionals and academics no longer need to be like 'passing ships in the night'.** It's time for both sectors to look for ways to partner and solve issues together.

Industry professionals also need to support the education of students or developers in the early stages of their careers. Some of the more experienced engineers will have learned the hard way and discovered some key secure coding principles by fixing their own mistakes. This knowledge – while not absolute – will be invaluable for junior developers.

Plus, mentorship programs can be an excellent way to nurture young talent while simultaneously preparing the future SDLC workforce, arming them with experience and understanding of security best practice.

Where next?

We're not yet at the point of mainstream adoption for proactive secure coding over the culture of fixing known issues and vulnerability patching. There is still a lot of work to be done before this is achievable.

To reach that stage over the next 12 months, industry and academia need to consider:

1 Better Communication

Promoting better communication across an organization will re-enforce the concept of failing safely. Human error is the biggest percentage of insider threat incidents. So it's essential to set up an environment within the SDLC where it is expected and OK to immediately report a mistake. Everyone makes mistakes, so a key part of secure coding education is learning from mistakes and understanding how to avoid them in the future.

2 Increased Investment

In academia, courses in application security need to become a priority. To make this possible, there must be buy-in from those teaching and those making the financial decisions around the curriculum. In industry, the situation is very similar. For security education to be a priority, it always comes back to budgeting and whether the board has visibility of the reduction in risk. Unfortunately, for some enterprises, it may take an increase in data breaches or a cyberattack before the management and leadership teams truly commit to making application security a priority.

3 Many More Resources

Finally, we need more resources that are readily available for the average developer. Everyone in the SDLC needs to have the information on file and courses on hand for when they need it the most, so it can be applicable in real world scenarios. It also therefore becomes a journey towards safer application development rather than a 'one and done' exercise. And rather than raising their awareness of critical security problems, teams gain the ability to proactively create secure applications by embracing the shift left.

With additional resources, investment, and communication, organizations stand a much better chance of moving towards more proactive secure coding practices. Collaboration across private industry, government bodies, and academia is integral to this progression.

Along with positive in-roads being made in the industry recently, the Security Journey Roundtable showed us that we still have a long way to go before insecure code becomes a problem of the past. The SDLC will continue to face significant pressure to deliver applications at speed, and these teams simply do not know what they don't know. The only way to address the root cause of the issue is with a continuous and programmatic approach to application security education.

SECURITY JOURNEY

HackEDU's spring 2022 acquisition of Security Journey brings together two powerful platforms to provide application security education for developers and the entire SDLC team. The two officially became one in August 2022 and are now Security Journey. Two approaches, one path to build a security-first development culture.

www.SecurityJourney.com | info@SecurityJourney.com

SECURITY JOURNEY