Title: Developing Secure Software: Universities as Supply Chain Partners
Transcript

Part 1: The Software Security Knowledge Gap

Julia Allen: Welcome to CERT's Podcast Series: Security for Business Leaders. The CERT program is part of the Software Engineering Institute, a federally-funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. You can find out more about us at cert.org.

Show notes for today's conversation are available at the podcast website.

My name is Julia Allen. I'm a senior researcher at CERT, working on security governance and software assurance. Today I'm very pleased to welcome Mary Ann Davidson, Chief Security Officer for Oracle. We'll be discussing the critical need to educate today's software engineering students about how to develop more secure software. So welcome, Mary Ann, glad to have you with us here today.

Mary Ann Davidson: Oh, I'm really pleased to be here. Thank you.

Julia Allen: So to get the ball rolling, how do you define secure software? And why is it something that business leaders, like yourself, are starting to pay more attention to?

Mary Ann Davidson: That's a great question. I think for a long time, and probably still in some enclaves, people think secure software is security software – meaning, you'll go buy a product that protects against "foo" attacks and you're done.

But I think of secure software as really a couple of things. Certainly, it is security-enabled software where you have appropriate security functionality embedded within the product. But beyond that, and I think this is really mostly the point of today's discussion, it is software that is designed, developed and delivered with security in mind. So it isn't just features and functions. It's the actual engineering and processes of the product so that it is resilient and aware of security threats, and takes appropriate mitigating measures within the software itself.

And why is this something people are paying attention to? Well, part of it is that, historically, software hasn't been something we thought of as critical infrastructure or IT systems were not thought of as critical infrastructure. But they are now. Almost everyone has an IT backbone to their business. And most elements of critical infrastructure – things that you think of, like, the telco's [telecommunications] and defense systems – and pretty much everything you can think of has an IT backbone. So IT itself is infrastructure technology and as such, it needs to be much more security minded than used to be the case.

Another factor is that more industries, in various ways, shape or form, are being regulated. And security tends to be part of that. So for example, I read a survey a couple of years ago that talked about the biggest driver for people buying security software, in particular, was regulations. That's what was driving them to purchase security products they hadn't already bought. Cost of ownership is an issue. I think, more customers are aware that they have life cycle costs associated with software. And one of the biggest ones that's unpredictable is the impact of having to patch systems, take their systems down, and go through all the testing. That's not something you can really budget for. You might budget for it in terms of the software license. But you don't actually know, over the life cycle of a product, how much extra resource you're going to have to throw at it to try to make it secure, by patching your way to security.

Julia Allen: So, Mary Ann, how did the need for – given that there's now a growing demand for more secure software in the ways that you described and it's showing up in just about every aspect of our lives– how did the need for software security education for today's graduating software engineers – or I think, in a lot of cases, a lack of it – how did this show up on your radar screen?

Mary Ann Davidson: Well on a personal level, I think, it had to do with a couple of salient points. One of them is I come from sort of a dynasty of engineers. I think everybody in my family for three generations has been civil engineer, mining engineer, something engineer. My father had a PhD in engineering and was an academic. So and I studied engineering.

So real engineering, actually, has this sort of approach, where among other things, you're expected that whatever you design is going to be resilient, sort of fault tolerant, not fail. And it's just grilled into your mindset. And when I started working in security initially, I would have discussions with developers where that mindset was not there at all. So that's, kind of one issue. It's like, "Look – your mindset isn't, in terms of safe, secure and reliable, and your personal accountability for this, but it is in engineering."

The second issue was my having to explain very simple things, which even I, as I admittedly a very bad programmer, learned in programming classes – like having to explain to otherwise very smart developers what's a buffer overflow and how do you prevent it? So I'm thinking, "Why didn't you learn this in your very expensive education? And why do we have to re-teach you, or in some cases, teach you things you didn't learn but probably should have?"

So those are type of the two main kind of issues. And it's only been accelerated by all the discussions recently about supply chain issues and software. Where does your software come from? What's its providence? And as a vendor, I say, "Well, part of our supply chain is the graduates we get from universities. So if we don't get a good supply chain product – a good security aware educated Developer – it's hard for us to produce secure software."

Part 2: Universities as Supply Chain Partners

Julia Allen: So what action did you specifically take to raise this issue with the universities that Oracle recruits from, and what happened?

Mary Ann Davidson: Well, actually my boss came up with this idea. I thought it was a good one. I wrote a letter to the top, probably the top 10 or 12 universities we recruit from. And I did this last year. So it was addressed to he department chairman of computer science, or whatever the equivalent was, and the dean of the, again, engineering school or whatever the equivalent was. And as nicely and professionally as possible, I pointed out that we recruit graduates from their schools. We spend, unfortunately, a lot of money fixing what I think are avoidable and preventable defects in software. Customers – the larger customer community, not just our customers – are concerned about the security worthiness of software to the point where it really is a national security concern. And I cited things like the DHS, Department of Homeland Security, work on software assurance. I cited the SANS coding certification as an example of the type of expertise I expected graduates to demonstrate. And lastly, I said, "In the future, we are going to change our recruiting practices to focus on universities who do place an emphasis on educating their undergraduates, and graduates for that matter, in secure software engineering principles." So it's like, "There's a problem. We're concerned about it. We want you to change your curriculum. And we're going to start making our purchasing decisions, if you will, based on that."

Julia Allen: And so then, you sent these letters out, and what kind of response did you get?

Mary Ann Davidson: Well, kind of a resounding silence. One university did respond – good for them, they're polite, so they get points for that. But their response was to ask us to fund – they were saying, "Oh, we are doing some of that. But we'd love you to fund grant money or something so we can develop a class." I'm thinking, well, okay. Universities are public institutions are pretty well endowed. And I don't feel like we ought to be paying them to do things they should be doing anyway. And lastly, developing one class is not going to solve this problem. I mean having every institution change their curriculum or curricula would actually fix this problem. So I do laud them for responding. But I got a deafening silence, which is disappointing.

Julia Allen: In a lot of our work, we've kind of observed that the most effective way to get this – build some momentum around this – whole software security field is kind of standard market pressures, supply demand like you had just described. You asking your supply chain, the universities, to respond to requirements that you have, because your marketplace is expecting it of you. And do you think you're just kind of ahead of the curve, as to why you didn't get kind of a more broader response?

Mary Ann Davidson: Well, again, I don't think the supply chain issues have percolated down to universities. But it's a good chance to stop and say I'm kind of focusing on the people who didn't respond. But there are, of course, a lot of universities who do think this is important who are doing good things. In fact, I was at a university about two weeks ago speaking. And a young woman from a particular institution said, "Yes. We've actually instituted the basic secure programming course. And we've rewritten all our other classes to include security principles."

So my take away, not just for her institution, but for others, is going to be to collect a list of institutions who we think are doing the right thing. And take that to our recruiting people and say, "Look. Maybe we're not going to totally throw over where we recruit from in terms of maybe the top five or six institutions. Maybe we will. But at the very minimum, we should add these institutions to the places we recruit from because they are doing the right thing." And I can't very well tell people, "Change your curriculum," and then not change our purchasing practices, if you will, or recruiting practices to reward the things that we say are important to us.

Julia Allen: That makes good sense. Have you seen any other kind of similar action or some kind of response from your peers in the software vendor community? – both in terms of their own commitments to developing more secure software, given the market pressures they're facing, and perhaps a similar outreach to the universities that they hire from? Are you seeing any traction there?

Mary Ann Davidson: Well I'm seeing a couple of different things. One of the main ones is, and this is kind of really sort of a happy happenstance. Now granted though, the people I'm talking to are self-selecting into groups that are discussing assurance. But it's true that when I go to various software assurance meetings or fora, the mainstream vendors I meet there are, by and large, when we compare notes about "what are you doing here and how do you handle that?" we're all doing a lot of pretty good things now. This kind of idea that no one pays attention to this is not really true.

So I think some things have changed to the positive. I think the next level is to try to rope in kind of the nimble innovative start-ups. Because again, you can build a wonderful widget. But if it's going to be used in a critical application, then it needs to be designed and delivered securely. So we need to broaden our outreach to smaller vendors.

In terms of the interaction with universities, I've heard a number of vendors say that universities will teach this if you give them the materials. So where are we? Well, I try to encourage broader tents of developing courseware, particularly in – I've been recently corrected about this. I thought that it was ACM that's the accreditation body for computer science programs and I've been told that's not true. It's a different group. So I need to go back to whoever the right group is and say, "But at the end of the day, the accreditation has to change to require security principles and practices as part of the curriculum." Because that actually lifts the bar for all universities, meaning you can't just send somebody through a computer science program and not learn this stuff.

Part 3: Reengineering the Curriculum; Practices to Deploy in the Mean Time

Julia Allen: You had touched on secure coding practices as one example. But are there some other kind of key principles and practices that if you were queen for a day and could design the curriculum that you'd like to see?

Mary Ann Davidson: It's very interesting. I've had two responses when I've discussed this. I actually had one professor say, "Well, I thought about teaching this. But really we're interested in teaching our graduates ideas. We want to be having them focus on ideas." I'm thinking, "I understand that. I don't think this is just mechanical." But engineers are very creative but they're creative within the bounds of solid structures, dynamics, and physics. And if you don't get that, you can be wonderfully creative and have your building fall down.

So if I were queen for a day, what would I do? Well yes, I'd reengineer the curriculum. There needs to be a basic secure coding class. And it needs to also be embedded within every single other class. What do I mean by that? If you're in engineering, let's say a civil engineer, you don't get to forget statics when you get to upper level classes. You're expected to remember basic physics of buildings. And if you design a wonderful creative building but it doesn't load properly and it would fall down, you get an "F." Simple.

Also focusing not just secure coding techniques. It's how do you think about security. So having like a security architecture class – whether you call it threat modeling or however you – it's your entire approach to security. It's not just how to validate input properly, which is technical. It's what is your overall approach to security in terms of how you design systems. And that gets back to – I do take the professor's point about ideas – using automated tools. Nobody thought that engineering went to hell in a hand basket when somebody came up with handheld calculators and we all replaced our slide rules. Boy, that really dates me, doesn't it? But I've actually heard – really I'm only 26.

I've heard tools vendors say, "Look. We've offered automated tools to universities, and they don't want them." Saying well that really is best practice in development. I mean if you have 60 million lines of code and you don't know what kind of automated tools are out there to find defects in software and you can't use them? Manual code reviews are not going to solve that problem. You can't get through 60 million lines of code. So those are sort of three things: the architectural approach to security; the technique of good secure programming; and being able to use the tools that are out there. Some of that is ideas and practices. Some of it is technique.

But what you really want – in my dream world, what I want is for every developer, or ever computer science graduate I should say, to come out of that program and understand, "My code is going to be attacked. I need to design, develop, build, and deploy defensively. If I don't have a defensive mindset, I'm going to fail." Their mindset needs to change. If I could get that mindset to change, I can declare victory and hoist the flag.

Julia Allen: So this is a nice segue into kind of my wrap up questions for you – which is, you obviously have a gap you've had to fill within Oracle. You've had to do things to make sure that your software engineers and developers are able to produce the kind of products that you're held responsible for being produced. So in addition to some of the other things you've mentioned, are there some specific practices, either at the engineering level or, say, at the project management level, that you've instituted within Oracle that has helped you kind of move down this path?

Mary Ann Davidson: We're doing a bunch of things. And a lot of these have been over a period of years, at least ten years if not longer. So it's not like, "Hey, we just woke up yesterday. Security's important. Let's have a big initiative." For example, when we develop products, we have templates for functional design and test specifications, part of every project. And security is already part of those templates. And not only part of the templates, but in the basic template, it links back to our secure coding practices standards — which are about 300 pages now including things that are case law, meaning we don't just tell people "don't do 'X.' Well, that's great. How do I do it the right way?"

So we have examples of — it's not just don't do "X." Here's how to handle this particular problem the correct way. We have release checklists. For every single component on a product build, somebody has to go through a bunch of checklists that we designed to make sure that they did the right things all along. And we track those. We report on them. We do use automated tools. Some we've licensed, some we've developed ourselves.

I have an ethical hacking team, for example, who, bless their slightly evil little hearts. Their real job is not bug finding because that's an expensive use of them — it's knowledge transfer to development. We have courseware on secure coding practice; basic courseware that everyone has to take in development. I mean everybody — product managers, release managers. We have more detailed classes. We seed development with expertise by developing what we call a SPOC program. No, not the pointy eared Vulcan, but Security Points Of Contact. So we have people of excellence in every development group that we train to be the first line of defense.

Metrics — I try not to be a metrics fascist. But one of the things that you can use to help figure out are we doing better or worse, and how well are we doing, is to actually keep metrics around things. And I try to be very sensible about this because I want to reward good behavior, not just "Oh, you reported a bug. Let's go shoot somebody." So you don't want to have disincentives. And really to me, if nothing else, it's a source of analysis. This development group seems to have more problems in this area and they have a harder time closing bugs. Why is that? Is it because they're all idiots or is it because half their team is on a maternity leave? Or maybe we need a different training vehicle. Maybe there's a new type of vulnerability. We're saying maybe our training materials need to change or our tools need to be better. So that's one of the things. Particularly in a large development organization, there's so many things you could focus on. If you don't measure what you're doing, it's going to be really hard to hone in on the problems that are the worst things you need to tackle that will have the highest payoff if you get them right.

Julia Allen: This is really interesting the way that you describe the different things that you've put in place. And so it causes me to want to ask you a follow up question, which is how do you distinguish the roles and responsibility you and your office and the practices you're trying to put in place with either the business units that are responsible for getting the product out the door or the project manager and the development team? Who owns what part of the space you've just described?

Mary Ann Davidson: Well, I think there are lots of discussions about how do you craft a security group — centralized, decentralized. My long-term goal is to be out of a job. Now that sounds really strange but development owns their code. We have to worry about many things when we pull out a product; performance, for example, portability. We don't have a separate portability group that focuses on how do we make our code portable. We embed it within development practice, process, reward structures, measurement. The only way we're going to succeed in security is not to be by my tripling the size of my team. It's going to be trying to continue to seed expertise and processes and tools so development can do – well, they already have the responsibility for security – but they can do it well and better. And then my team might have a governance function. "Okay, let's just make sure we actually are doing these checklists correctly." But I don't want my group to be the be all and end all in security because then we will have failed.

Julia Allen: The idea is to get the software security practices integrated as part of the normal software development life cycle, right?

Mary Ann Davidson: That's exactly right. And we try to do that. And to me it's like what's effective? How do we make it easy for people to do the right thing? Now there are people, I will tell you, most people are pretty good. But the ones that I really do want to ride herd on is not somebody who made an honest mistake in development, right?, okay? "I really try to check input conditions but I made a mistake." It's the people who willfully do not think this is important. Those people need to be run out of town on a rail because they poison the entire development group.

The people who try, and make an honest mistake, we can help that. We can figure out how we prevent that in the future. But if somebody willfully doesn't comply then it's basically – not to be too Draconian – it's cure them or kill them because this is important. It's an important cultural value. It's important to customers. At the end of the day, I've actually gone into development groups and say, "You know what? You're responsible for U.S. national security." "What do you mean?" "Well, you're not responsible for all aspects of it. But we have lots of customers in all sectors. All of them keep secrets. Some of those are national secrets. So you know what? You are responsible for the security worthiness of every single line of code you write. It's not QA's job. It's not my job. It's not the ethical hacker's job. It's not release management's job. It's your job.

Julia Allen: So, ultimately, turning this into a cultural norm sounds like what you're up to.

Mary Ann Davidson: That's what you have to do. Engineers – you never discussions about "Hey the bridge was really elegant but it wasn't designed to stand up under loading." Right? You just don't have those discussions. They understand it can't fail. With developers, it's still a cultural issue. "Well, this is an elegant technical solution." "I'm sorry. You know what? I don't really care. I want to know if it solves the business problem and does it well. And it does it with strong security worthiness. Otherwise, elegant technical solution, that's interesting, but it's not enough."

Julia Allen: Well, Mary Ann this has really been a great conversation. I think, raised a lot of very valuable points that will be useful for our listeners. And so to wrap up, are there some places that you'd recommend where people can learn more about both software security education and the right thing to do?

Mary Ann Davidson: Well, there are a lot of really good books out there. I'm happy to say, in the last five or six years, one of the big changes, there's been an awful lot of books written about this. And that's a good place to start to do some web searches. One of the other things I would be remiss if I didn't mention was the DHS Software Assurance Forum. It's a website called Build Security In on CERT, U.S. CERT. And I think part of the answer to this, by the way, isn't just individuals doing better, it's contributing to large community discussions. Because frankly, while there are some institutions that are doing very, very good things here, bless their hearts, we need to raise the water level for everybody.

Julia Allen: Well, Mary Ann, again, thanks so much for your time and expertise today. I really appreciate it.

Mary Ann Davidson: Well, it's really a pleasure. And I look forward to seeing the world change and everyone developing more securely. I'd be really happy to be out of a job because this problem goes away.

Julia Allen: And onto the next.

Mary Ann Davidson: Exactly.