

The Silver Bullet Security Podcast, episode 21: A Panel Discussion with Cigital's Principals

The Silver Bullet
Security Podcast
with Gary McGraw



Listen to this show at <http://www.cigital.com/silverbullet/show-021/>

sponsored by



Gary McGraw: This is a Silver Bullet security podcast with Gary McGraw. I'm your host, Gary McGraw, CTO of Cigital and author of Software Security. This podcast series is co-sponsored by Cigital and IEEE Security and Privacy Magazine. For more, see www.computer.org/security and www.cigital.com/silverbullet.

This is the 21st in a series of interviews with security gurus, and I'm pleased to have here with me today some of the principle consultants from Cigital. Gentlemen, please introduce yourselves.

John Steven: Hello, this is John Steven. I'm primarily interested in helping companies build their own software security capability.

Pravir Chandra: Hi, this is Pravir Chandra. I do a lot of work in training, along with helping our customers do strategy all the way down to security assessments.

Sammy Migues: Hi, I'm Sammy Migues. I own a lot of the service line management here at Cigital and do a lot of monetizing of intellectual property.

Gary McGraw: So, John, we'll start with you. You've been involved with a number of high-profile enterprise software security programs for customers like Fidelity, FINRA, and Morgan Stanley. What makes for a successful enterprise initiative in software security?

John Steven: I think that people have to remember that there's multiple prongs to any software security attack, so to speak, on their organization. They have to make

changes to some of the process pieces and the way their organization develops software. They need more knowledge, whether that's hiring new people, or training their people. In addition to process, they need a knowledge component, as well as there are some technology components, both in terms of products that they can use in concert with their own applications, as well as tools such as Fortify that they can use in the assessment of their software that they have to bring to bear in order to be successful.

Gary McGraw: So what's the best way for a big company to get started with an initiative in software security?

John Steven: There are a lot of things that people are not doing, or doing wrong with regard to software security, but there is no best way to start canonically. I would say that people need to start with what they do well, their strengths, and focus on those. So if you have smart people focusing on giving them a little bit more security knowledge, playing to their curiosity is helpful.

If you have a strong central governance capability in your software development organization, putting in place standards and forcing teams to adhere to those standards may be more effective for you. It really depends on the kind of organization you're in.

Gary McGraw: Can you give us an example of one of your customers who's started a certain way? I was thinking maybe FINRA would be a good one.

John Steven: Well, organizations start various different ways. I would say some of the common ways to start would be adopting a static analysis tool or beginning with penetration testing. Other organizations have focused on beginning with training to raise core awareness within the organization and seeing what gaps come up with after that.

FINRA is an interesting example because they started with trying to get a handle on their pile of applications, what they have and where the risk in that portfolio of applications is present. And that allowed them to – will allow them to, moving forward, pick their security, where they spend their security dollars most effectively.

Gary McGraw: So Pravir, you were instrumental in creating CLASP, one of the predominant software security frameworks. And CLASP and Microsoft's SDL and Cigital's Touchpoints that I wrote about have more in common than people might realize. What common ground do those three approaches share?

Pravir Chandra: Well, I think that they all are largely in agreement about the kinds of activities that you need to be incorporating into how you're building software. I think the ways in which they differ tend to be more along the lines of ordering in which you might go about it, incorporating those things into your process, or even just the extent to which you go about doing those activities, like there's various different levels to which you could scrutinize code, for instance, and to find bugs.

So I think that's probably where the differences are, but the commonalities are really pretty strong because most of them are just talking about, or the three at least that we're talking about now, between CLASP, the Touchpoints and Microsoft's SDL. They all incorporate elements of reviewing the different artifacts that you have at different stages such as requirements, SDL slightly less on requirements, less so than with Touchpoints or CLASP, all the way through to reviewing code or reviewing the

finished product where SDL is a lot heavier with the security push that you have at the end, whereas CLASP and the Touchpoints go about it a little bit more of a balanced threat, the lifecycle fashion.

Gary McGraw: So do all three of them, are they all able to address whatever software development lifecycle you may have in your organization already?

Pravir Chandra: Yeah, I think so. And I think any of them really could. The difference is really what they're best suited for, I think. So for instance, and I've talked to lots of our customers that have expressed the same opinion, things like Microsoft's SDL is very well-suited to an organization that would be operating as an ISV, where they're building and selling this packaged software, the components that they're putting together, whereas for a lot of, say, financial institutions or retail where they're actually building and then operating their own software, something like SDL is not really well-suited to that environment just right out of the gate. You have to do a lot of modifications to it yourself, if you were going to try to follow the strict regiment of SDL. So I think things like the Touchpoints or CLASP are more suited to the different breadth of organizations out there and the ways in which, different ways in which people use software.

Gary McGraw: So if you were trapped on a desert island with a whole bunch of software developers, sorry about that metaphor, what is the one touchpoint or best practice that you think is the most important that CLASP and the Touchpoints and SDL all share in common?

Pravir Chandra: Wow. So I guess in days past, I probably would have answered code review. But I think my current thought on it is actually just doing risk analysis of what – so Microsoft calls it doing the threat modeling phase. CLASP calls it something in between threat modeling and risk analysis, and the Touchpoints call it risk analysis where you really just need to know what exactly the value is of the application to

your organization in order to really even determine how much effort you should be putting into the whole review process anyway, because they're definitely seeing people spend way too much money on applications that couldn't possibly have cost them as much as they spent on security. And I've seen quite the opposite, where people are spending almost nothing on security for an application that's their whole core business.

So really, just getting yourself situated in the right direction is probably the most important thing to start off with.

Gary McGraw: Sammy, you've been in information security for multiple decades. What role does training play in security, and is software security training any different than the other kinds of security training that people have been doing in network security for a million years?

Sammy Migues: So the answer is, the training is very different. The way I've seen training change over the years has been a large focus on awareness training in the beginning. Awareness training still serves IT security rather well. We just need people to be aware that you shouldn't let people trail behind you when you're walking through the door. We need people to be aware that you shouldn't click on links in mail notes unless you know where it comes from. We just need people to be aware of a lot of things, and they probably won't do it.

When we need someone to actually do something, though, when we need someone to be a practitioner and not just an enabler of bad events, we need to give them a different kind of training. We need to actually show them how to do it. We need to give them the tools to get it done, and we need to help them understand the context of when they should actually do this work, how much effort they should put into it.

So when we give people software security training, we need to actually tell them how to implement what we want them to do.

We need to give them a skill, not just make them aware. So for me, that would be the biggest difference between software security training and traditional IT security training.

Gary McGraw: So, does it require a different sort of instructor when you're doing that?

Sammy Migues: Ah, good question, absolutely, so your instructor for software security training I would say almost universally has to be a practitioner. The instructor has to be able to go off the script, has to be able to go beyond what the training material actually says, and be able to answer questions for the students. There's pretty much no chance that all the students will be at the same level of knowledge, the same level of ability, the same level of training. So the instructor's going to get a lot of questions that, let's just say, will vary widely in the level of understanding by the student. So the instructor's going to have to be able to take it from the beginning and give an answer to the student that they can actually use. If they're not a practitioner, they stand virtually no chance of doing this.

Gary McGraw: I think one of the other risks is that developers tend to pay attention to people like themselves and not pay attention to others.

Sammy Migues: That may be the case. I'll have to say I don't do teaching in front of classrooms of developers, so I can't answer that. But I've certainly done teaching in front of classrooms of pen testers, IT security geeks and other people who think you have to be one of them to teach them. Or you have to be smarter than them, and you have to prove it, so you have classes where the first 20 minutes is stump the monkey and all that kind of nonsense.

By and large, though, if you prove that you know what you're talking about, and this is something that's valuable for the people to learn, I think you can win over most crowds and the curmudgeons and the others who want to play stump the monkey all day, you

just – little teaching techniques to marginalize them, put them aside and let the other people actually learn.

Gary McGraw: Yeah, guns work pretty well. I saw you laughing over there, Pravir. Have you been subject to stump the monkey?

Pravir Chandra: Oh, yeah, all the time. I do a lot of the teaching for classrooms of developers. And, yeah, that happens. That happens pretty often. Although usually by the time you hit about 10:00, and you've done your first break, you kind of answered most of their questions and convinced them that you're pretty smart, and so then they listen.

Gary McGraw: So back to John. John, you recently wrote a Justice League posting about threat modeling. What's the difference in your mind between threat modeling and architectural risk analysis?

John Steven: I think that's a good question. Threat modeling, at least as Microsoft has written about it in their literature and their books seems to be a combination – well, really, an activity that developers and QA people engage in together. But it combines things like STRIDE, a canned list of conceptual security attacks like spoofing or tampering with data flow analysis at a very low level.

And in my mind, when you're doing an architectural risk analysis, the conceptual notion of threat modeling is an important start, figuring out – but as we at Cigital have talked about that, what we mean by threat modeling is a threat is someone who's attempting to break into the application, and they will use attack factors to try to – to access the application. Sammy's actually going to follow up my posting, if he hasn't already, with some threat modeling vocabulary to disambiguate some of this.

In my mind, for practical purposes, if you're trying to boot either architectural analysis or threat modeling in the your organization,

figuring out who's going to attack your system is a good first step, figuring out what assets within the application are worth protecting is a good second step. And then brainstorming what kinds of attacks people may use to get access to those assets as objectives or intermediate objectives such as session tokens and some of the common things that people target are good first steps towards a threat modeling exercise.

Gary McGraw: So there's one other first step. It might be step zero, if you're going to be really geeky about it. And that's, in order to do proper threat modeling exercise, or proper architectural risk analysis, you need to have an architecture.

John Steven: Absolutely.

Gary McGraw: Or some way of having documented your architecture that's comprehensible. In the old days, it was difficult to find organizations that had that. But I think it's becoming more common for people to have better architectural documentation. What's your experience with that?

John Steven: Hmm. So my experience with that is, is you're right. That is step zero. You need to know what it is that your – that the threats, or that the threats will come against and the assets sit inside. And I think people are getting better at that, but my experience is directly that when you create the diagram and annotate it with things like threats or assets, they go, "Wow, is that really what the software looks like?" And I know you've experienced a lot of that in the olden days of Cigital as well, doing assessments.

Gary McGraw: Well, in the really olden days, and that's a term that Sammy really likes. You can see it on his face. In the really olden days, people would come with a network architecture, and they would say, my application lives on this machine, which is right behind this firewall doohickey, and that would be their architecture. I think we've progressed beyond that pretty significantly.

John Steven: Well, yeah. I think there's some implicit cheating involved though. Like when people were building apps with C back in the day, the programming language is like copper wire. You can build absolutely anything with it, whereas nowadays, when you're building an app on like a J2EE platform, there's so many aspects of architecture that are just enforced upon you by the platform itself that it actually makes it a hell of a lot easier to put together a document of what your architecture does because you're just following the patterns that have been laid out before you.

Gary McGraw: Mmm-hmm. That's a good point. So, John, how can somebody develop expertise in architectural risk analysis, or threat modeling?

John Steven: I think that first and foremost, they have to have experience, and I'm talking about years and years here, not six months, in developing software. And I think that that experience has to include a leadership of development teams in at least capacity of low-level design. If that hasn't occurred first, then it's going to be very hard to make a lot of progress in an architectural analysis.

The next step, I would say, is really looking at the software and arguing about what it is supposed to do for the business that it supports. And often times, I think what we've found here is that even if you don't have – even if you don't come to the table with a huge tool kit of esoteric security attacks that you can check against the software, being able to argue about what it is that the software structure and behavior looks like, and what it's intended to do, will very quickly uncover either ambiguities or omissions. And what the team, when they designed or developed it, they ended up having weaknesses that are exploitable by an attacker.

Gary McGraw: Yeah. One of the things I like to do to get at that is to lock two software architects in one room with one piece of software, and surely, some big fight

will develop. If you just videotape that, you can often find interesting places to focus your attention.

So, Pravir, you've been active in a lot of community groups like OWASP and The Shmoo Group. What role do communities like those play in security?

Pravir Chandra: Wow. Well, I think they play a pretty damn important role. Overall, just thinking about the influence that a lot of these organizations have had, like for instance, OWASP, the OWASP top ten, take it or leave it as to whether or not you like it or you don't like it, but it was instrumental in what actually went into versions of PCI, which became what the companies around the globe are actually jumping to be in compliance with.

So I think that as time has gone on, I think that groups of smart people that are interested in solving some of the bigger problems in security just kind of tend to naturally congregate, and that's what these groups are in general, like especially OWASP, or The Shmoo Group or any number of other open groups that are out there, like OWASP and so forth. They're just people that have an idea of something that they can do to kind of make the world a better place, and they get like-minded individuals together, and pretty soon, you have something that's kind of independent of corporate influence that is trying to push some good idea forward, or something that's good in general for security or privacy or what have you. So, yeah, I think they're pretty powerful.

Gary McGraw: Sammy, with the acquisition of WatchFire by IBM's Rational Group, we may see more emphasis on security testing as practiced by QA. So how much of the security testing burden can or should QA take on? And do you think QA is going to end up taking over pen testing, in the long-run?

Sammy Miguez: Well, it depends how long the long-run is. Jumps are how high, sir? It really is going to – how much money

an organization is going to be willing to spend on technology to enable people who are traditionally not technical engineers to be able to execute technical engineering tools and make the results useful. And the answer is, they may be willing to spend a lot over time, but it's not going to happen any time soon. And then most of the security tools really rely on – how shall we say – categorizing and implementing known vulnerabilities.

So in the sense that QA should be using automation to find known vulnerabilities in software before it leaves the organization, that should be happening now. Any organization that puts software on the Web, that has an easily found, scriptable known vulnerability is just making a huge error. And they should be using QA to find that. We do not need technical experts to find that.

Gary McGraw: Wielding a tool like SPI Dynamic's tool or WatchFire's tool, or Cenzic, something like that.

Sammy Miguez: Absolutely. And also taking a look at the software after it's in operations, which is another important aspect of this, typically not QA's job, but there's a lot of vulnerabilities that we can introduce through the general network configuration that may not be discoverable by the individual QA person. But that's off on a tangent.

So getting QA to the point where it can think like a bad guy, have a notion of an existence of a vulnerability and follow the shiny object down a rabbit hole is going to require an integration of security aware testers, scripters and other people into the QA organization, and that's the event that really needs to occur.

Will organizations sense this and start to do this? I think the answer is yes, and I don't think it'll take that long because they'll figure out that one security savvy person in QA is probably a lot less expensive than an army of consultants coming in on the backend and doing IV&V.

Gary McGraw: Yeah. One of the issues is going to be trying to figure out how to turn the information that comes out of those tools naturally into something actionable from a fix-it perspective because the tools today, I like to belittle them as badnessometers, but then I go on to say that everybody should have a badnessometer because they don't know how bad they are.

So let's assume that the golden age of badnessometers is upon us, and people start piling up the problems, surely, there's going to come a time where fixing those problems becomes just as important as identifying them in the first place.

Sammy Miguez: Absolutely. We certainly see that in our practice, where there are people who need help turning tool output into actual bug reports for software developers to fix. This is a niche, and I don't think the tools are going to be able to do that because even though it can take you to the line of code, it still can't tell you what to fix.

Gary McGraw: So now for something completely different. John, what book are you currently reading?

John Steven: Well, I'm dealing with a caved-in dining room ceiling, so I'm not actually reading a book right now. I'm in between a –

Gary McGraw: A fix-it book?

John Steven: Yeah, I tried that, which is why the hole is as big as it is right now. I'm in between, actually, A People of Paper, which you recommended, was outstanding, and Ordinary Wolves, which you've also recommended.

Gary McGraw: Good, good fiction reading.

So, Pravir, how does it feel to own a car that's made almost entirely of titanium?

Pravir Chandra: Not bad. It goes real fast, so that's always a good thing.

Gary McGraw: It sounds kind of dangerous.

Pravir Chandra: Yeah. It's good. It's a good thing.

Gary McGraw: And Sammy, what is your favorite video game?

Sammy Miguez: Wow. Gee, right now, Call of Duty 4. I don't know. That's really tough. I would say so.

Gary McGraw: Okay. Well, happy holidays, everybody. Thanks, guys, for your participation.

This has been a Silver Bullet security podcast with Gary McGraw. Silver Bullet is co-sponsored by Cigital and IEEE Security and Privacy Magazine where Silver Bullet excerpts are published in every issue. Subscribe today at www.computer.org/security/bsisub. Show links, notes and an online discussion can be found on the Silver Bullet Webpage at www.cigital.com/silverbullet. This is Gary McGraw.

[End of Audio]