Securing Open Source Software in the DoD

Featuring Scott Hissam as Interviewed by Linda Parker Gates

Welcome to the SEI Podcast Series, a production of the Carnegie Mellon University Software Engineering Institute. The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts.

Linda Parker Gates: Welcome to the SEI Podcast Series. I am Linda Parker Gates, and I am the initiative lead for Software Acquisition Pathways in the SEI Software Solutions Division. Today I am joined by Scott Hissam, who is working on software assurance in DoD [Department of Defense] systems. And Scott recently put together an event on free and open source software and wrote a blog [post] about the event, and that's what we're going to chat about today.

Welcome, Scott.

Scott Hissam: Hello. How are you? Doing well, thank you.

Linda: Great. I am fine, thanks. Nice to see you. Can you tell us what you're doing here at the SEI these days? What you're working on?

Scott: Well, I have been at the Software Engineering Institute now for a little over two decades. Since that time, most of my research has been mostly primarily interested in component-based software engineering and integration of those, mostly focused around not only functionality but also the quality of the software itself. That also includes the security of that software, which is a key quality attribute. All that should apply to what we design and build and buy. From my perspective, it really doesn't matter if we are talking about proprietary software. Proprietary software is a type of software you would get from say a commercial company and even open source software. I like to call that free and open source software or FOSS.

Linda: Yes, so let's start there. Can you explain what we mean by free and open source software?

Scott: You have to go back a little bit in time and have this sense about where open source software came from. It has been around, I would say … Oh my, I should have done the math ahead of time, but 70-some years. There are a couple authors who I know in Ireland, Feller and Fitzgerald, they did a book on open source software. They traced the beginnings back to 1953 in a project called PAC. I can't remember at the moment what PACT stands for [Project for the Advancement of Coding Techniques]. There are others tracing the sharing of software back to 1955 with IBM and a project called SHARE, where they were trying to actually share software between companies and scientists that were using IBM mainframes. But it really wasn't until 1998, kind of going through the '70s and the '80s, when a person by the name of Christine Peterson actually is the one attributed to coining the phrase *open source software*. The reason that this was done was because up to this point, the term of art seemed to be *free software*. That was a little bit of a misnomer. It was basically conflating two concepts where the real source or the real sense here is that open source software conveys the idea of freedom: freedom to do, freedom to understand, freedom to learn. Not so much free as in, well, the quote is *free as in free beer*. I typically will use the word FOSS for free and open source software to underscore and convey that freedom and openness of the sharing of ideas in software.

Linda: Cool. I always wondered about how that word *free* should be interpreted given that it is right up against the word *open*. I think it's an underscoring of how the software can be used. So the government, DoD organizations, are using FOSS. Can you talk a little bit about how they are using it and how frequently they are using it?

Scott: Well, OK. The federal government and, in fact the Department of Defense, the largest agency, they are not really different than anyone else now in industry or even folks that live at home. Pretty much whether you know it or not—and I am not speaking to you specifically, Linda, but to the community at large—there is a lot of FOSS software, free and open source software, that is being used by practically everybody, from television sets to your internet of things to some major banking systems and major systems around the planet. The DoD is not immune to that. A lot of the software that is used in some of the core of the operating systems they use, even if they are from proprietary vendors, are benefiting from what has been learned over the past, well, 70 years of exchanging free and open ideas and free and open source software among each other. So the DoD finds itself having to deal with free and open source software all the time.

Linda: Well, so specifically then, in the defense industrial base, the DIB as we call it, what is going on there?s going on there?

Scott: There is a little interesting story. I think it is important to bring this through. In the mid-1990s, there was a gentleman by the name of David Wheeler. He at the time worked for one of the other FFRDCs, federally funded research and development centers, in the United States. He

was at IDA [Institute for Defense Analyses]. He was able to convince the legal community at that time that according to the defense federal acquisition regulations [DFARS], that open source software or FOSS was, in fact, a commercial item. It was a commercial item because in the federal acquisition regulations, anything was considered a commercial item if it was licensed.

In the FOSS community, software is, in fact, licensed, but they fall under what is called free and open source licenses, like the MIT License, the Apache License, and even GPL [GNU General Public License]. There are a number of licenses well beyond what I just listed that are approved and recognized by the Open Source Initiative as free and open source licenses. Given the fact that FOSS is a commercial item, this brought FOSS to the same level as what is called proprietary software. That is also a commercial item. Commercial companies produce proprietary software. The Department of Defense gets into the situation of being able to not only recognize but also use FOSS in acquisition of basically DoD mission systems. I think the first salient memo that I remember was in 2003. It was called the Stenbit Memo, and it pretty much outlined this exact point. Over the last two decades, the DoD CIO office has been providing clarifying guidance on the use of open source software in DoD systems including in 2009 and most recently in 2022. This puts open source software on that same level playing ground with proprietary software.

I think it is also important to note, going back to that free and free beer comment, this does not conflate or include the concept of software that is also known as *freeware* or *shareware*. Freeware and shareware, this kind of software sort of dominated the landscape in the ’80s and ’90s and ended up becoming a vector for vulnerabilities for infecting organizations. A lot of this freeware and shareware is, in fact, binary based. They are binary-based executables. There are artifacts. It’s impossible to vet them. You don’t have the source code to them. You have no idea what’s in them. I like to be careful and separate the FOSS, free and open source software, from these malicious and malignant types of freeware and shareware that used to permeate the landscape.

Linda: Right. You mentioned guidance coming from the DoD CIO’s office, can you tell us more about that?

Scott: Well, actually, I think it was the most recent memo, the 2022, and I’ll talk specifically to that one. I thought it was a very exciting memo. The author from that office, his name is Dan Risacher, the way that he lined this out was along three perspectives. One was from the consumer perspective. One was from the producer perspective. And the third was from the collaborator perspective. Let me walk through that just a little bit.

From the consumer perspective, it takes I think a very healthy perspective; it says focus on adopt, buy, create. And that was a mantra that was laid out in that memo.

The first idea is, there is a step in acquisition which considers something called analysis of alternatives. In that analysis of alternatives, where you are trying to make choices between competing products, the memo wants to make it clear that FOSS is, in fact, a shelf in the marketplace that you can pick from as an alternative. The memo wants the DoD community to adopt FOSS when it's possible. If there is nothing available in the marketplace that can be adopted from the FOSS community, then the buy. That's where they adopt [then] buy. Then the decision comes down to buying something from the proprietary or commercial portion of that marketplace. Finally, if you can't adopt and you can't buy, then you build or create. That is not uncommon given the DoD mission. The whole point here is to try to avoid creating new. If you are creating something, try to avoid creating new versions and adhere to open source licenses. That is from the consumer perspective.

From the producer perspective, when we can't adopt, we can't buy, and we create, the mantra now is, *Make it open by default*. The guidance from the DoD CIO office is, where possible, the software that you create should be open by default and should be released as open source software, recognizing that what is released has to be nonproprietary and disclosable to basically the world. There are certain exclusions about what that means to be disclosable or not in the memo.

Finally, the third point is *collaborator*. In this perspective, it's not so much that the DoD is producing a new component or even consuming a component. But what they want to be able to do is be able to collaborate with the FOSS community to contribute new code. The memo makes clear that government civilians, military personnel, and even contractors can contribute to FOSS communities while they are on duty, as long as it's in the benefit of that DoD program, and for contractors, if the contract allows it. Probably the most important thing with collaborating is, avoid trying to make DoD-specific version and forks in a void? So when you are contributing code to the open source community, you work with that community, adhere to their processes, their review processes, and their release processes so that your changes can be folded in to those upstream processes.

Linda: Yes. That is really good. I know there is a downside, though, because there always is. So what are the caveats, Scott, around using free and open source software?

Scott: I am going to go in reverse; so now from that, I just talked about the collaborator. From that collaborator perspective, while they are working with the open source software community, they are reminded that they need to maintain operational security. What operational security means for any civilian, military, or [contractor] working on a DoD program, is really specific to that program. They have to know what that means. They have to be able to maintain operational security. For me to say what that actually means for any specific program, it's hard to be general about.

In a producer standpoint, we tried to adopt, we tried to buy, now we are creating. So now we are producing, we are doing it open by default and releasing that open source code. The key here is that when producing and releasing that code, you have to follow disclosure processes. There are ways of disclosing information from a DoD program to the general public. You have to follow those processes.

Linda: So the government is doing something about these risks, it sounds like.

Scott: Well, they are, in fact. I think the other interesting thing about a producer I want to say is that they also have to be in a position as the producer to be the maintainer. They have to have the budget, and they have to have the ability to manage its enhancements from the community back into those within the DoD that are actually producing that open by default.

I think the last important one is the *consumer* role. This is probably the biggest problem of that consumer, producer, collaborator role. You can do all these things to adopt and use open source software in the DoD as a consumer, but you have to be able to manage supply-chain risk. You have to be able to manage software assurance. That is where my research and my interests right now are. In managing that supply-chain risk, when you are consuming FOSS, you have to be cognizant of long-term support. Using trusted sources—that in itself is a challenge. You have to deeply understand the dependencies of the open source software that you are bringing because those dependencies have their own dependencies. You have to make sure that the software is vulnerability free, and there are techniques for doing that, of course. You have to be cognizant of malicious actors in the supply chain.

Linda: You shared an example with me the other day about an analogy to bolts. Do you remember?

Scott: Right. Yes. Yes. So here up to this point we've been talking about software and open source software and software supply-chain risk. When I think about software supply-chain risk, I often like to use the bolt analogy. And the bolt analogy is, there are rules in the DoD that, when they use a component, it doesn't matter if it's a processor, an engine, or parts of an engine, or a bolt; they need to know that, when they start using that product, that they can expect to be able to use that product for very long-lived systems. For me, when I am going to a home-improvement store and I am getting parts for my deck— So let's just say I am buying a bolt to bolt my deck to my house— I am just not going to buy any bolt. I am going to buy a bolt that I know can hold the deck to the house and not collapse. So you are looking for a specific bolt size that has certain strengths and certain characteristics.

The DoD has to do that. They do the same thing. But they also say, *Well I am also buying this bolt from this company, is this company itself known to use good practices? How do they ensure*

that the products that they produce are, in fact, quality products? And how do I know that? And so they say that they adhere to certain testing standards, and we see those kind of things all the time. The same thing is what the DoD is trying to do now with managing supply-chain risk for software, not only from a security standpoint and vulnerability free, but also to know that the software itself has a certain level of quality, it's been tested, and the community or the producer that is producing that software is, in fact, going to be around for a long time.

In the Linux community—folks listening to this may understand—there are free and open source software distributions of the Linux operating system with its desktop environment. Some of these companies actually give you guarantees. They say, *If you use our Linux distro, we will maintain it for so many years. We will maintain security updates, sometimes even longer than that.* I feel, when I use that desktop environment, I can rely on that particular distro to give me assurances for the next 5 to 10 years that I will continue to receive security updates as long as I have that particular version before I am forced into an upgrade situation. I hope that makes sense.

Linda: Yes, it does. There was recently a presidential memo, I guess May 2021, about improving the nation's cybersecurity posture. How does that relate to this work that you've been doing?

Scott: Yeah, it's called the Executive Order on Improving the Nation's Cybersecurity. I think it's Executive Order 14028. And basically it called for a number of organizations and agencies to improve their cybersecurity posture for the nation and its critical infrastructures. And there was direct guidance—I don't have all the details of that—but there was very direct guidance to NIST to securing the software supply chain for the United States. NIST has completed that work. The result was two key documents. One is called 800-128 [800-218], which is Secure Software Development Framework. The other was a guidance on software supply-chain risk management. Both of these were in direct response and responding to the order from the president at the time.

What I think is fascinating in that is that, although there was very specific guidance given in there for the supply chain, the NIST material was actually building upon, well, to NIST's credit, over a decade's worth of prior NIST standards, from configuration management, security configuration management, supply-chain risk management, and even cyber supply-chain risk management. All those documents were updated, pretty much in response to that presidential memo.

Following that memo, the Office of Management and Budget [OMB] just released in September, Memorandum M-22-18, I think, or 1218. I can't remember now. I think, yes, it was 22-18. And it was called Securing Supply Chain Through the SSDF [Secure Software Development Framework]. There is the reference back to NIST on the Secure Software Development Framework. Those just came out. There is lot in there, and that is OK. But it basically came

down to two big important pieces. One is directing and requiring DoD agencies, or actually I think it was Title 31 agencies, to start using software bill of materials. Think of a software bill of materials as being the ingredients in something you buy. It says, *Here are all the components that make up the thing you just bought*.

Linda: That Ikea parts list that you get?

Scott: Yes, exactly that. It comes with its own problems and its own risks, but the guidance is clear. The Department of Defense and the defense industrial base and the producers that produce software to provide software bill of materials for the software that you acquire from them. Then the other aspect that is important in the OMB is an attestation statement that the producer that produced the software you got it from attests to following rigorous standards for software development. The details are in the memo, and what that means for attestation. Now, again, for the work that we are doing, we are trying to roll those, what does it mean for a DoD organization to actually consume these attestations? What does it mean for a DoD organization to consume these software bill of materials, and what do they do with them?

Interestingly, that memo came out in May of 2021, like you identified. The audience on this podcast may remember that the famous Log4j fiasco came out in that fall. There was a number of hearings. There was a number of critical infrastructure implications of the Log4j hit. Given that focus on that executive order, there is now a new bill making its way through Congress now called the Securing Open Source Software Act of 2022. Put a pin in that, and we'll see how that comes out.

Linda: I want to bring us back to the SEI, though. Relative to the DoD, we are a small FFRDC, a federally funded research and development center. What is our role in the FOSS landscape? What are we doing?

Scott: As you know, and maybe not everybody knows, the Software Engineering Institute at Carnegie Mellon is the only federally funded research and development center that is chartered to look at software — software processes, software technology or software techniques, software processes. We don't necessarily differentiate between free and open source software and proprietary software; it's software. But recognize that there are differences when one is beginning to think about software assurance and when thinking about software supply-chain risks when it comes to open source software. My job recently has been trying to get insight into those projects that are actually contributing and producing in open source software, which is actually being used within the Department of Defense. You mentioned at the beginning of this podcast, we had a workshop. The blog [post] for that workshop is listed at the bottom of this podcast. One of the things that was very important coming out of that workshop was the concern the DoD had at the highest level, but also from the folks on the floor developing software, is

being able to get insight into objective evidence, the health and stability of some of these open source software products. Then how do we understand, how do we get insight to what they do? As explained in the blog, there was a very clear message—and I appreciated the message that we heard—was that buy-in and trust the process of the open source software community. That in and of itself, although it is wonderful to have that and I want to be able to do that, I need to have that evidence that tells me that what I think is occurring within that open source project actually is doing that. The community, the FOSS community heard that concern loud and clear and gave us in the workshop, not only myself, but some of the DoD folks that were attending there, pointers and hints to tools that would actually help us get that objective evidence into those concerns.

Linda: So you convened that workshop back in June as an *un-conference*. Can you tell us more about that and some of the key outcomes?

Scott: This, in my mind, is a huge topic. You can look at this from so many different perspectives that it almost made my head spin. The idea of bringing, first off, *Who do I bring together? Who should come together?* I make it sound like it is me, but it is not. I was just convening the discussion. But, *Who should come together, and what are we going to talk about?* And so I worked with … I started small. I had a few contacts in the FOSS community. Those people knew people, those people knew people, and those people knew people. So you've got that network of network of network. I would like to say this happened by design, but it didn't happen by design. It happened this way. What we ended up getting was a number of different foundations that wanted to come and talk about the topic. The topic was really about how the DoD would get its hands around FOSS from a supply-chain software-assurance perspective.

So a number of the foundations showed up, folks from the defense industrial base. More in the cloud area came. Some DoD organizations came. Some federal organizations came. We now had a critical mass to begin that discussion. Rather than me thinking, *Well, this is what I want to talk about,* the idea was then, *Let's take it from an un-conference perspective and just start talking and then just start identifying topics.* I think over the course of the meeting, we generated about 12 different topics, about a dozen topics or so. In that un-conference style, we all used a voting scheme to say, *What do we think we really want to talk about? What should we try to tackle first?* Lo and behold—and maybe this happened by accident or maybe it was subconsciously by design—but the top five I thought were really perfect for my immediate needs. And they were: trust the open source software and the FOSS processes over any specific individual. Risk management when dealing with the consumption—that is that consume, produce, collaborate perspective from the DoD CIO memo—but the risk-management perspective of consumption of FOSS. Actually getting hands-on supply chain artifacts from those communities that are producing that software. Then how do we keep these structures, how do we keep the discussion going between those that were in the workshop and the community around it? It was from those

discussions that vectored me off now in doing some very specific—well, experiments is a strong word, but trials of ideas. I am in the midst of doing that now.

Linda: So what is next?

Scott: If things go the way I hope they go, I now have a framework. I hate the word *framework*. I don't know what else to call it. But I have an approach that I think may actually give us that objective evidence that is necessary to understand the community that is producing these products. I have done some internal reviews of this. I am now slowly working these reviews out to others in the community, not only our stakeholders within the DoD, but also some of the individuals that were actually in that workshop. I am hoping very soon to be able to broaden that lens on that work soon.

Linda: I know you are working on ways to bring the results of all of these conversations back to the community. I know that is a big part of why this is important.

Scott: For me, I think the most important thing is, I really feel I got a lot of great information from that first workshop. I want to give back, OK. I want to be able to let them see that not only that I listened, but the DoD side of that conversation listened. This is what we are trying, and what were the things that seemed to work well? Where are we still struggling with being able to get that objective insight and trusting that these communities in these projects are doing the things they say they are doing?

Linda: So there will be a link to the blog [post] in the transcript of this session. And I am looking forward to … I can recommend the blog already because I've read it. And I look forward to what else comes down the road from this work that you're doing. And so I just want to say thanks for talking with us today. And, again, we will include links to the transcript and to the resources that Scott mentioned. And, finally, unless you have, do you have something else you wanted to say, Scott?

Scott: I don't think so. I thank you very much for giving me the opportunity to let folks know more about this and hopefully leave your comments in the blog.

Linda: Right, yeah. As a final reminder to the audience, our podcasts are available on SoundCloud, Stitcher, Apple Podcasts, and Google Podcasts as well as the SEI's YouTube channel. And so if you like what you see there, please give us a thumbs up. And thank you very much for joining us today.

Scott: Thank you again. And thank you for listening.

Thanks for joining us. This episode is available where you download podcasts, including SoundCloud, Stitcher, TuneIn Radio, Google Podcasts, and Apple Podcasts. It is also available

on the SEI website at sei.cmu.edu/podcasts and the SEI's YouTube channel. This copyrighted work is made available through the Software Engineering Institute, a federally funded research and development center sponsored by the U.S. Department of Defense. For more information about the SEI and this work, please visit www.sei.cmu.edu. As always, if you have any questions, please don't hesitate to email us at info@sei.cmu.edu. Thank you.