SEI Webcast

Modeling DevSecOps to Reduce the Time-to-Deploy and Increase Resiliency

by Joseph D. Yankel, Aaron K. Reffett, Nataliya Shevchenko

Shane McGraw: Hello and welcome to today's SEI webcast modeling DevSecOps to reduce the time to deploy and increase resiliency. My name is Shane McGraw, outreach team lead here at the Software Engineering Institute. And I'd like to thank you for attending. We want to make our discussion today as interactive as possible, so we will address questions throughout today's talk. So, you can submit those questions in the YouTube chat area and we will get to as many as we can. Our featured speakers today are Aaron Reffett, Natasha [sic] Shevchenko, and Joseph Yankel. Aaron is a senior engineer in the SEI's security automation systems group; while Natasha specializes in systems engineering, model-based systems-based engineering, and threat modeling methods, all within the CERT division here at the SEI. Lastly, Joe is a senior engineer in the initiative lead of the DevSecOps innovations team here at the SEI. Now, I'd like to turn it over to Joe Yankel. Joe, good afternoon. All yours!

Joseph Yankel: Good afternoon. Thanks for the introduction, Shane. And, for everyone out there watching, I really appreciate you coming in. Please hop on chat. Let us know where you're from. I know we have viewers in the United States and all across the world. So, we're excited to hear you in chat. All right, so, a little bit about what we're doing here.

Joseph Yankel: We're all highly involved in DevOps and cybersecurity and modeling practice in software engineering. And, so, we get to work with many programs across the DoD and one of the things we're seeing is that programs are really struggling to implement a DevSecOps strategy. So, we have a little graphic here showing what we're seeing. We're seeing, basically, program offices are playing whac-a-mole, trying to guess where they should focus their efforts. They're struggling with overspending on tools, on things that they don't really understand. And what we're trying to do here is we're trying to help develop a strategy to allow these programs to understand exactly what they need to develop a DevSecOps system, tooling, the infrastructure, and the people needed to build and program. And that's what we're going to discuss today. So, we're going to talk a little bit about some challenges and I'm going to introduce Aaron, here, to go over some of the things we're seeing here and the things we're working on.

Aaron Reffert: Thanks, Joe. I appreciate it. So, as Joe mentioned, our perspective at the SEI and for this initiative is we're focusing a bit more on government program offices. All three of us have extensive experience working within DoD program offices and there's unique challenges that each of these offices are encountering in attempting to implement DevSecOps. So, as we see here on this slide, standard DevSecOps' infinity loop that probably most of us are very familiar with it-- it emerged out of industry and, at this time, government and DoD is attempting to try to integrate it into parent [ph? 00:04:43] software development cycles, methodologies, and so forth. But DevSecOps isn't just technology and that's something that a lot of folks initially trip over. So, they say, "Okay, well, if I have a software pipeline and

some tools, and I kind of tie them together; and I put software in at one side, I get codec and execute and run out at the other side. And I'm done, right? I've done DevSecOps." It's really a lot more than that. It's-- the focus is on people, their roles, and the processes. You know, you don't [ph? 00:05:17] technically need to automate much to do DevSecOps if you've got people and processes. So, we like to think of this and kind of back out to give it-- it is a socio-technical system that is augmented by software, but it's primarily people who have to write the code, who monitor the systems, who operate the systems on a day-to-day basis, and are working with end processes. And those processes ensure that all of these stages flow neatly from one step to the next, that the correct software is being written, that's it's free from vulnerabilities, that it meets the needs of the end users, that operates within various security controls, environmental constraints, and so forth. And adopting the correct set of software in order to augment, that is the goal. But that's a lot harder than meets the eye, particularly there's no one size fits all. And I think that's the crux. There are hundreds of pieces of technology that could be adopted and configured and combinatory in infinite numbers of ways that can implement a valid DevSecOps ecosystem end-to-end. Which ones do you pick? Which ones meet your needs? And, so, that's what we're trying to model here. We're viewing this as not a thing to be acquired. This is more of find the right people, adopt the right processes, choose the right technology, acquire the right technology. But that's a small part. And then put it together in a way that fully meets your needs. And next slide.

Aaron Reffert: Appreciate it. So, connecting process, practice, and tools. So, starting from business mission-- I'm trying to word this in a way that is not necessarily DoD specific, but what is the goal of the program? What is being acquired or built? What is going to be delivered to the war fighter? Deriving business cases and requirements from that, but then it gets split out into two sides: capability delivery on the left-hand side, products on the right-hand side. And if you look back at the infinity loop, the left side is typically development, the right side is operations. But when viewed as a whole, simply the ability to deliver an application into production is a massive capability unto itself and can be developed independently of the actual applications that an end user might see. And that might be, and probably is, far more complicated than the applications themselves, because automation-- it's harder to automate something than it is necessarily to do it manually. Could be an order of magnitude more difficult to have a fully automated end-to-end DevSecOps system with automated checks and gates and so forth. And all the infrastructure that's built on, all the platform tech, whether you're on AWS-- I'm just throwing things out here. We're not-- you know, _____ 00:08:43 the right choices or not, you could put Kubernetes on AWS or you could just use AWS with normal images and all the various services that it provides. And getting that set up before you even get an application put onto it, is an undertaking to itself. But the program office is responsible for overseeing all of this. It's not just the thing that's being built. It's too easy to focus on, "Well, here's my business case and requirements for the thing that I'm building that's going to be delivered to the war fighter," and focus on that, and lose sight of the larger chunk of the iceberg that is sitting under the surface of the water, which is that platform, infrastructure, shared services, and so forth.

That needs to be developed to the same effort that you would put into developing the products themselves. And, so, yeah, I think I mentioned the processes, practices, and tools. It's developing all of that. My personal interest right now, and the work that is backing this, is a lot of enterprise modeling, model-based engineering. And you look at that infinity loop, there's no start or end. It keeps going. But how do you get into it to begin with? What do you do in order to adopt-- what stage is zero that gets you into this feedback loop that you can then continue to build and iterate on? That's an area we found tends to be kind of glossed over in the standard literature that you'll read about DevSecOps. It's, "Oh, well, you get all these technologies, you put these processes in place, and bing!" It just-- you know, you're off and running and you're creating great code and all is hunky-dory.

Aaron Reffert: But we'll see in the next slide here, being in the CERT division at the SEI, I really care about the security. I really care about the security. So, even if you've acquired all these processes and tools, technologies, so forth, how do you know it's secure? And that's one aspect. That's just cybersecurity and there are software assurance is another big part of this, too. But you're either just focusing on the cybersecurity, how are you sure that-- I have all these pieces in place that, a, I've picked the right ones, and, b, how do I know they're secure? Within DoD there's-- I've got my 853 controls that I need to implement, all the various overlays that my programming required; how do I know whether this vast Rube Goldberg machine that I've built actually has all the controls and in the right places? The controls, they're going to be replicated. It's a system of systems. So, the same control will apply, potentially, to all parts of your system. How do you know where they apply and how they apply? That's just part of this modeling effort, is to-- and when we get to Natasha later, she'll talk a little bit more specifically about kind of how we will map these various requirements constrained onto model elements. And, so, we can point and say, "Okay, at a platform independent level, this is where you care about these various things," and do so in a methodical way that allows a program office then to down-select and say, "okay, here are the things that I care about and here are the things that apply to me and here is where I need to address them." And then moving that model forward into that platform-specific model, which is actually representative of your system. So, that's the goal of kind of where we're going in cybersecurity, is one aspect of it. But software assurance is another big piece. And cybersecurity can largely be seen as the right side of the infinity loop; software assurance is the left side of the infinity loop. But you need to do the two together. And, so, how do you actually know whether you're doing the right things or not? That's kind of laying the foundations for kind of what this work is. And the next _____ 00:12:54, we'll talk a little bit about kind of how the tools are selected, fit together, kind of how that methodology works and we'll get a little bit more into what the enterprise architecture approach that we're taking is.

Joseph Yankel: And I think those are really good points that you mentioned there, also, Aaron, that this is, like I said, a socio-technical-- we're talking there's a lot of people involved. Just this infinity loop, it's expanded, right? Now, we know we have business mission, right? So, in acquisition, we've got to talk,

"Hey, how do we acquire this? We've got to think of bigger picture. I need to acquire things in a way that allows me to do some of these DevSecOps ideals." So, it really changes the landscape of what we're looking at here. Not only do we have to do design an appropriate pipeline to enable all this automation, but I need all the different experts to chime in on their expertise on what's actually required here.

Aaron Reffert: Exactly. Absolutely. You're looking at a program officer's traditional role, is program management, project management, oversight, contracting, acquisition with enough engineering support to be able to evaluate deliverables. How do we expand that now into a world where a program office is being an integrator of a software-centric system? They're taking on the role of development and, so, instead of saying, "Okay, well, we're bringing a prime and a prime will be responsible for the end-to-end software development life cycle and fulfilling all these things. All we have to do is feed the requirements, we'll evaluate what they're doing along the way to make sure that we're getting what we expect." Now, there are government or uniform civilians and military personnel writing code directly within these various programs. And, so, you've torn away that layer that a typical, ordinary prime contractor, prime integrator would do. And, so, now, all of those skillsets, which kind of get blurred between-- your prime will say, "Okay, we'll bring all these skillsets." It's all in their RFP. They'll tell you what they're doing. Now, all of that has to be done by the program office. And that's a monumental shift. And how do they do that?

Natasha Shevchenko: I would just support what Hugh Boss [ph? 00:15:21] said that approaching this issue from different point of view, from the different roles point of view, actually kind of dictate enterprise architecture approach, because enterprise architecture allows you to build these different points of view on a system, allows you to incorporate voices from different pieces of the enterprise, which is <audio cuts out 00:15:57> become essential to enterprise, incorporate them into a system, not [ph?] building, and show it as one coherent piece not as separate elements, not as a style law [ph? 00:16:13] that tried to solve their own local programs. And I hope we will be able to show it a little bit later in our slides.

Joseph Yankel: Excellent. And, so, we have some slides up, but we just kind of want to talk about the fact that these systems are becoming extremely complex. They're very hard to even hire the skillset, to understand what you really need, and some of these slides just show that. But there's more and more smaller pieces that do different functions to maintain these things and to administer them, to appropriately secure them, it's a monumental task. There are services out there that are providing some of this. And they're expensive.

Joseph Yankel: But if you try to build your own, you'll probably quickly find out that it's very expensive to do so. There's a lot of expertise. Really solid DevSecOps processes and pipelines and toolings is expensive. It's large. It's very technically challenging to understand all the moving pieces. And, so, we've put together a few slides just talking about really basic interactions with a very simple system. And we've

put together some metrics here on just the couplings. And this is, again, just a simple system. Here we are with, you know, over 30 different interfaces between tools.

Joseph Yankel: So, the idea, what we're trying to say here, is just this is hard. This is really hard to roll your own. And it requires you defining appropriate roles, right, in DevSecOps. The folks you need to set your agendas, right, to decide, "What do I need for my program in place before I begin this," right? Before I design, before I buy anything, before we build, there's quite a bit of planning that needs to happen and then some hiring that needs to happen, right? There's the idea either we're going to determine where our talent is or go find that talent. And that's a challenge. And some things that we're trying to take into consideration as we try to model this is identifying the critical roles, the ones you need to have, and those roles will be responsible for filling in the rest of the gaps there, as far as the folks we need to do this stuff. But, again, systems, they're new, they're more complex than ever. We keep on throwing new technologies at this that help alleviate some things, but what we really want to do is define a lot of the core tools and what they do. So, our approach will be, "Let's talk about what we need to do, and then we'll choose tooling and people based off of that approach."

Joseph Yankel: And, so, I don't want to hang onto too many of these tools in particular, but the idea is this is very complex. It's a challenge. It's been a challenge for programs we work with to hire for this. It's been a challenge for us to really understand the big picture itself.

Joseph Yankel: So, let's go into how we're addressing these challenges.

Natasha Shevchenko: Thank you, Joe. So, as was mentioned a little earlier, that we're dealing here with, first of all, system of systems. We're addressing big and complex systems that support development and aggregation [ph?] of the systems that provide capability to the customers. And these force us to look at that from a systems-thinking point of view. And we can't ignore that pieces of DevSecOps actually interact with each other, like we just saw a minute ago; that interaction is complex, that it's involves both people and technology, and it should satisfy big numbers of requirements. So, we essentially have another system that we need to build, we need to analyze, we need to support, and it requires specific tools. We can't stay with just scripts. There's style laws [ph?] and pieces here and there. We need to approach it as one coherent system. So, that's why we decided that it's time to address it from an enterprise architecture point of view and create a _____ 00:21:26 platform-independent model. Another aspect: Why we want to do that? Because if we have a model, if we involve digital engineering in our processes to develop our DevSecOps systems, we can apply system engineering processes to that. We can analyze systems. We can analyze the requirements' traceability. We can analyze gaps between requirements and capability and _____ 00:22:02 capabilities and implementation. We gave the people who need to build this complex system tools to make sure they build what they need to build and they

won't miss anything and they make sure that the system-- what, in the end, will do what they actually want it to do. And, in this case, I'm not talking about the system that provide capability directly to the customers. In this case, I'm talking about DevSecOps system. Next slide, please.

Natasha Shevchenko: So, I won't talk much about that, because this slide-- you will be able to read later-- is just showing that why it's not there. So, there is no anything similar to what we tried to do. There is a documentation that describes what should be done, very extensive, a lot of pages, a lot of words; and none of that is talking about how to do that. Another type of information usually people find when they look on Web [ph?] how to build [ph?] your DevSecOps, you have specific solutions. You have a sales pitch from companies that give you very specific solutions for the use case they described for themselves earlier. And this use case is maybe not exactly aligned with your use cases. Even it's looks very nice and they're very good in their job selling stuff, but if your use case is not aligned with the initial use case of the tools, there's very often the case you won't get what you want to. You will need to do some fixes, some adaptations of the tools, which essentially will do something but not what you want. So, that's why our goal is to provide platform-independent model that will describe the problem and give you tools to solve this problem specifically for your environment, your application to deliver for your organizational search and so on. Next slide, please.

Joseph Yankel: Natasha, we had a couple questions from the audience. We probably could answer them now. One was that many programs seem to have a hard time with the cultural changes necessary to begin to think about this. Old habits are hard to break. Is there any-- do you have any comments on that?

Natasha Shevchenko: I would say that the best solution for that, it will be gradual changes. So, you have your organizational structure right now and you get the indication that to get where you want to, you need to make your tweaks in the organizational changes on your processes. It means that you will need to create a plan of gradual change. So, yes, in the very beginning, you will need to make, actually, the final decision, "I want to go-- I want to make these changes and get my organization into different state." This is most important thing is to understand that you need this change and commit to this change. And then build a plan how you get from current situation to situation where you want to be. And, actually, models, digital engineering, and MBSE [ph?] allow to do that, because you can create model of your organization now, model of your organization as a goal in transitional stages. And these will allow you to analyze: Is it good for you, will it solve your problems, and how to better address this transformation? Otherwise, you will literally sit and just speculate, and it will be more like hand-waving than real solving the problem. See it in front of you in a formalized way, it will help you, definitely, to address this issue.

Joseph Yankel: Yes, I know there's pretty much a demand that we try to do DevSecOps. We know of many benefits, but, in our model, we are identifying critical roles. And one of those roles is your DevSecOps champion. This is a role that must be had in a program for the person driving the real goal, you know, the goal of DevSecOps. So, I believe culturally it really starts with "There is a champion with this. We need others to believe that." So, there's definitely a big cultural change. But I do think it's a change that we're going to have and we're going to probably learn along the way the best approaches for that.

Aaron Reffert: I'd also like to point out that this is not just a problem with existing program offices. That brand new program office operate-- developing entirely software-based systems also suffers this problem. And it's because of the longstanding culture within the government and DoD. Experiences on traditional legacy, waterfall-type programs perpetuate themselves into the next program as the person, during their career, moves from program to program. And, so, a lot of those things that worked on those older types of programs, those habits get carried forward. So, how do we address that? And that has been a challenge that we've recognized, of even when you're starting out from the very beginning, greenfield development, how do you not carry forward some of those <makes air quotes> "bad" habits. I don't want to call them bad; there's plenty of experiences you can carry forward, but how do you adapt those to actually work within a DevSecOps world.

Joseph Yankel: Sure. I think what we're doing here is our model's describing this ideal state and part of the ideal state is to get capability into our users' hands quickly, and not just capability, but capability that works. And, so, what that looks like in a perfect DevSecOps world is that our end users are getting to see what we're doing early on in the process, right? And it gives them an opportunity to give us feedback: Is the thing we built, is it working? Because a core thing is we need the users to give us feedback so we can iterate, right? We need to be able to make these changes iteratively, securely, and provide them real value. So, if I'm looking at a waterfall program, I'm saying, "Can I do that? Can I get a small change my user needs to them in an efficient manner?" Right? So, I think the model potentially could expose the situations of the current state that don't allow that to happen. And, so, with that, you might have a transition plan, "What are the things we need to do to become more agile? What are the things we need to do to allow our users to get the changes they need faster?"

Natasha Shevchenko: The model definitely will allow you to analyze and see if you have bottlenecks of your processes, for example, or you have unequal distribution of automation during your processes, which, in fact, a whole pipeline, so-called DevSecOps pipelines. Because if one step in your process is highly automated and everything happened, literally, overnight, and then another step is highly manual, your process will be as fast and agile as your manual process. And it's hard to see if, for example, your system or your DevSecOps is described in a document, in a Word document. There is no visualization,

there is no validation of the model. It's just human language, which can describe the same thing in ten different versions and everybody will understand something different on it. So, it's not analyzable, really. And, especially, not analyzable from a point of view of repeating the analysis, repeating the same analysis over and over and kind of cleaning up your model, tweaking it and make it more efficient, more suitable for you. You can't do that with a document. Every time analysis will look different, it will be different people applying different rules to this analysis. If you do it during-- using the digital engineering, it allows you to formalize analysis as well as just description of your system. So, this is probably couple of-- on the slide in front of you, you see a couple of points I did before why enterprise engineering. Enterprise architecture was used before-- actually, it was used in building the system of systems that provide capability to the customers and it includes building the processes around programmable systems. So, it allows you to model not only behavior or computer system, but behavior of your human system and analyze it the same as a computer system. Even if it looks a little bit strange, but it is possible. The main point is that you formalize description of your system using the common model language [ph?]. This is what happened probably-- this is hardest part in transition from document-based models or document-based architecture of your system to the MBSE, model-based system engineering, and digital engineering approach to solve this problem. When you need to formalize, you need to follow the rules and standards of modeling language to describe your problem or your use case, to describe your system in hand, and to describe your goal, where your system need to be. Technically, this is the goal for MBSE. Next slide, please.

Natasha Shevchenko: Yeah, this is what we're talking about. So, we work on creating a PIM, platform-independent model, that will describe the problem of DevSecOps and describe the behavior of DevSecOps system independent from the solutions completely, independent from a platform where and how it will be solved, but it will give users of this PIM all requirements, all high-level architecture relationship between elements of this architecture, analytical tools to analyze your system, and including the computational system as well as your organization, your human system around the DevSecOps. And, after that, the user will be able to create platform-specific model, to create a solution for the problem described above, during PIM modeling process and then, using their platform-specific model to implement this model into real life, into real system. So, we really hope that the PIM will give all support for the new existing programs to get this slip, to jump from waterfall, jump from separate siloes, pipelines-represented DevSecOps into the world where your DevSecOps-- it's coherent system allowing working all of these pieces together in harmony and analyze it and make it modifiable, make it more evolvable. So, if any changes need to be done to this DevSecOps system, you can simulate these changes in your model first and then implement it if it's needed. Because very often the changes may need tweaks and it's better to do it on a model than on a real system. First, it will eliminate a lot of risks that can get your implementation expensive and not working for you. Next slide, please.

Natasha Shevchenko: So, these are some-- this is a diagram we came up with. It's very high level. I tried to explain how this PIM will work. So, in the very beginning, you have your PIM, you have requirements, standards, planning. And, out of these, a platform-specific model will be created. And having a platform-specific model, then you will build your actual DevSecOps system and deploy it. And, next steps, you will run the analytics of your system, collect the data, and then you will figure out if your system needs to have a change. I believe Joe can add some more details on this diagram, specifically, and the different boxes on that.

Joseph Yankel: Yeah, so, at a high level, this describes what [sic] a platform-specific model may be used. Right? So, what we're saying here is that all the requirements in an independent model should be able to produce the systems, right-- and by "system" we mean kind of the DevSecOps toolings, processes and people, right? This is your system. This is a system that you would then develop an application on. Right? What we're trying to describe what a DevSecOps system is, right? So, people generally think of continuous integration, continuous delivery. We're saying it's a little bit more. It's the people, right? Your platform-specific model defines the roles, it will help you define the tooling and the infrastructure, and it takes into consideration the-- you know, the applications, the software or hardware that you do have to support, but this DevSecOps, in this picture, is really describing the machinery around producing an application. Okay. And, so, what we're describing here is the fact that, in theory, in the future, a properly configured platform-specific model could potentially generate you configurations to deploy a system. Right? So, there are some really easy use cases. If you were strictly a cloud-based system, micro-services, this might be achievable today, right, to model a system that could be deployed. But many of our needs require very specific use cases. So, the idea here is we want to model the system we need to provide to develop an application. Not just that, but we want to take into consideration the ability to gather the appropriate metrics, right? And, so, this describes a few of the ones that we care about in terms of resilience, right? What's going to allow my system to be more resilient, make sure I'm secure? So, we've identified and we're working on identifying more metrics of infrastructure, of cyber, different constraints you might have, human risk elements, ML-AI model risks that you might have in the actual machine learning training pieces of it. So, we have many different divisions working on different aspects, but a DevOps system needs to be able to gather all the metrics of relevance. And, of course, our end users: Their feedback is a metric that we need to consider so we can, in the future, be able to make automated changes, right? So, this model, what it's accounting for is the fact that we have small changes. We have a box here called "iterative changes" that describe changes that our system is capable of handling, of correcting and re-deploying, for instance, right? And then we also describe changes that are larger architectural changes. So, these are metrics that our system might provide to expose things that we didn't consider in our model. And, so, what would happen there is that that now feeds our model to provide us a new specific model. So, there might be real use cases out there today. The SolarWinds attacks comes to mind, that there was a system that did not do something, right? That means that system wasn't built to

handle that event. Our model would then be updated to make sure that we are building systems capable of doing those things. So, this is kind of a high level look at how you might use this model, what it means.

Natasha Shevchenko: We have a question. Thank you, Raj [ph?]. You are asking, "Are there any suggestions to scale down proposed DevSecOps approach?" Can we answer this question right now about scale down? I think if you will have a model, you can actually figure out which piece is suitable for your use cases, which one is not. And it will be kind of obvious for you, especially if you look into roles that organizational structure of your DevSecOps system need to perform. And for big programs, big companies, it may be departments that perform this role. And in the small scale, it may be one person wearing multiple hats. But model will tell you that in any case, you will need to perform and perform these roles.

Joseph Yankel: Yeah.

Natasha Shevchenko: It doesn't matter if it's one person or a whole department. Go ahead, Aaron.

Aaron Reffert: Yeah, scaling down is difficult, because you don't want to cut out anything that's critical. And I think that's the key part of what the model will reveal to a program that adopts it, is you can't get rid of things that must be done. So, the lower bounds of what you can scale down to will come out of the model. And, so, yes, people-- you can scale down number of people if maybe _____ [ph?] isn't necessary, people can wear multiple hats. But the model will see, "These roles need to be fulfilled, these processes need to be implemented, these constraints need to be met." That's your lower bound. And, so, that gives the program a little bit more assurance that they're adopting the right things and only the right things. And when they're developing a minimum bible product or standing up a program to begin with, that's what's important is you don't want to do too much too fast. That's something that we've noticed in some of the programs that we've worked is they attempt to adopt too much too fast: Try to adopt everything, do everything from day one. And it rarely works as smoothly as the programs would like it to be. But if they had a good grasp of what that minimum viable was, what the lower bounds of their scope should be, we feel they would have been able to execute on a much more narrower set of features and functionality, and then be able to build themselves up.

Natasha Shevchenko: Okay. There was another question. Thank you, Anthony. "Will any of the previous work performed by the SEI to develop reusable MBSE architecture families be integrated into the DSO?" DevSecOps. I think this is a goal. We're just starting. It's out first effort to create any model regarding DevSecOps. So, the future iteration of this work will be to incorporate other work from SEI, and our main person, actually, to guide us there it will be Joe, because he did a lot of work in this area. That's why he is actually part of our project, to guide us into the DevSecOps solutions he and his team developed before.

So, we will integrate it into our model, so our model will have these answers to these specific use cases for these specific questions that was actually answered already. So, do you have any additional comments for that?

Joseph Yankel: I can say it's been a real challenge and the creation of the model has actually been a bit of a blessing, because it's really identified the things we've glossed over before, right? A lot of us jump to "Let's implement this. Let's use this hot new technology out there that's solving all these problems," but almost in every case, without a real, proper design, we've missed it, right? We've overlooked something. The model's really helping us enforce at the requirements level and at the roles level the people that need to be involved to make these decisions, they're involved. It's a key element to this. It's really an enforcement of all these ideals of DevSecOps, how it comes together is describing it-- I mean, there are lots of guides out there. The problem with those guides are they're very specific. They're specific to a type of application, most of the time and then we spend a lot of extra effort adopting it. We're trying to curb that and understand the specific need for our program, right, on what do we need to build to support what we're trying to do? And how do I make sure, at the end of the day, I've built a thing that's meeting our need? Right? I need to get this feedback. So, our model's trying to describe the system that'll eventually feedback the right information for us to improve it. Right? So, what we're describing here-- and we are building some prototypes of this, so, everyone in the SEI can-- we have experts inside our threat-malware analysis. We want to try to provide an exemplar platform that demonstrates this. Because we want all this different expertise to come into play and help us out. And, so, our challenge was "Let's get this started so we can have others collaborate with us."

Natasha Shevchenko: Yeah, it's excellent point, saying that the PIM, the platform-independent model, should be suitable for different kind of system analysis, including threat modeling, cybersecurity analysis, and so on. So, in some way, you can look at a PIM as not only the model of a potential system, but also as a tool to create a system, to analyze a system, to understand what your system needs to look like, how it need to behave. And, also, it will actually play a role of conveying right type of information for different type of people and still describing the same, coherent system. So, you can have different views and still see that it's one system. It's not siloes. So, again, we try to marry different aspects: DevSecOps, system thinking, MBSE and digital engineering, and, of course, like, system engineering practices in general. All together. I think DevSecOps, in general, radiated to this level to be a system by itself. Joe?

Joseph Yankel: Okay, so who cares? I think everyone cares. The struggle is real. We are trying to help. What we hope comes out of it is that we get some information, we get some folks willing to take a look at this and see if it helps them identify some of the questions we get. "Can I do this? Can I do that?" We're hoping that this helps answer the questions of "Here's what we might need to meet this ideal that we're doing better." And, also, we're really taking a look at the fact that we want-- we have to understand and

monitor our own system, right? Not just the application, but the system that builds the application. It's a big attack surface as we've recently discovered. It's vulnerable to errors. And, so, we're trying to describe the appropriate system that enables the development of secure software. And I know we're down to about ten minutes. So, we'll take any other questions that we have, but this is the ideal. Let's build something that's more than paper guidance that leaves a lot of room to interpret. We're trying to basically _____ 00:51:47 organizations to self-assess their own needs and give them the guidelines to do so, to understand what things we need in place, you know, what type of system we need in place that allows us to securely develop code, that well tests code, that allows us to know that this code can operate at the end environment and we're very comfortable with that, that everyone that has a say in the deployment and operation of a piece of software or hardware has made that say. And that's it. So, really, we thank everyone for your time. Any questions?

Aaron Reffert: Yeah, I'd like to circle back to a question that came in earlier. Mark Printa [ph? 00:52:30] had a scenario for us and I think this is a good time to look at this. So, "How do you transition to DevSecOps practices given the following common scenario? A legacy, waterfall program organized around functional disciplines which [ph?] are protected, but largely monolithic software using the, quote-unquote, 'right tools'-- for instance, Jenkins-- but with minimal true CI and CD automation." I think there's a lot going on there. And this is a very common question that comes into us and one of the first that gets asked by legacy program offices. And there's a lot to unwrap here. And I think the first one that jumps out to me, personally, is that DevSecOps is not necessarily the-be-all-and-end-all of software development. And it may not be the right approach to developing software for you right now. But that doesn't mean that you can't adopt-- for instance, there are software architectures that are more amenable to DevSecOps than others. Because, remember, DevSecOps is not just the type of software, it's how you operate it. If you're trying to ship the Titanic every two weeks, it's going to be very difficult. And just the time of testing and verifying that you didn't break anything in a monolithic system like that is really going to slow you down. You're probably not going to be able to deliver that quickly. But if you're able to break it down into manageable pieces with well-defined interfaces and ship those smaller pieces more rapidly, you'll probably find yourself in a situation where DevSecOps might help you out.

Joseph Yankel: Right. Typically, you know, we do an analysis. Right? We understand these ideal situations. We want-- we talk a little bit about this-- we want to be able to quickly develop and get changes to our system. So, I think in a typical waterfall, the environments work. We're actually doing the testing appropriately, we're comfortable with our end product. But there's some pain point-- to do something maybe faster, right? And, so, what we want to look at is here's these ideal concept DevOps that say-- they say, "I want infrastructure's code. I want well-tested software. I want users to be able to let me know that what they're getting is good." Right? So, this user interaction, user experience, is really important. So, these concepts, we take a look at our own situation, say "Our we able to deliver on these?" If not, how do

we improve that? So, I think it's all individual. It's different for every organization, but that's what you'd look at. You'd say, "Okay, these are these ideal concepts. Which ones can't we meet? Which ones bring us the most pain?" That might be the way we focus on that. It doesn't mean I have to re-architect everything right off the bat. But I might be able to bring some pain down.

Natasha Shevchenko: Yeah, I agree. Asking right questions from the very beginning is extremely important. If your current DevSecOps works for you, there is no question it's ideal, maybe you don't need to change anything. If it's not identified where it's not working and I would try to preach-- again, if you will have the example of the DevSecOps model in front of you and you can map your specific system element to the PIM and see how it's holding and see if you missed anything, or you don't miss anything, and can map a PIM actually help you to analyze your system and find the break point or points to improvements, identify where more automation can happen, should happen. So, make this analysis of your system more formal, more repeatable. Right now, anything we want to repeat the same way over and over, so we can have sustainable and similar results. So, again, if you will have more formal representation of your system, your DevSecOps system, you will be able to apply the analysis on it over and over again. And this is one of the actual goals, to be able to verify and validate your system. This is one of the ways to do that. So, it may be even your architecture is a waterfall. Maybe this is what happened, especially if there is more hardware involved. And, traditionally, any hardware development, it's waterfall. It's very hard-- it's extremely hard to implement Agile or any iterations when you have hardware development-- like, production of hardware. So, maybe, actually, you need to have more or less waterfall process, but, anyway, to have a way to analyze your system and try to improve it, this what we are trying to do with the PIM.

Shane McGraw: Aaron, Natasha, Joe-- great discussion today. Thank you all very much for sharing your expertise.

Joseph Yankel: Thanks for hosting, Shane. It was a pleasure. And, anyone, please reach out. We're happy to answer questions. We like to make sure we're addressing some of the needs of the folks we work with and the rest of the world out there.

Shane McGraw: Great. And we'd like to thank everyone for attending today. Upon exiting, please hit that "Like" button below your video window and share the archive, if you found value. You can subscribe to our YouTube channel by clicking on the SEI seal in the lower right-hand corner of your video window. And, lastly, join us for our next live stream, which will be on March 18 and the topic will be "DevOps Enables Digital Engineering" Hasan Yasar and David Shephard. Registration information is available on our website now. And we'll email that out as well. Any questions from today's event, please send email to information@SEI.cmu.edu. Thanks, everyone. Have a great Day.

Natasha Shevchenko: Thank you.

VIDEO/Podcasts/vlogs This video and all related information and materials ("materials") are owned by Carnegie Mellon University. These materials are provided on an "as-is" "as available" basis without any warranties and solely for your personal viewing and use. You agree that Carnegie Mellon is not liable with respect to any materials received by you as a result of viewing the video, or using referenced web sites, and/or for any consequence or the use by you of such materials. By viewing, downloading and/or using this video and related materials, you agree that you have read and agree to our terms of use (http://www.sei.cmu.edu/legal/index.cfm).
DM21-0031