Shane McGraw: Hello. Welcome to today's SEI webcast SolarWinds Hack: Fallout, Recovery and Prevention. My name is Shane McGraw, outreach team lead of 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 and you can submit those questions in the YouTube chat area. We will get to as many as we can.

Our featured speakers today are Matthew Butkovic and Art Manion. Art is the vulnerability analysis technical manager at the CERT Coordination Center, part of the SEI at Carnegie Mellon University. He and his team coordinate complex vulnerability disclosures, facilitate the discovery and handling of new vulnerabilities, and influence practice, standards and policy. Matthew is the technical director of the risk and resilience team within the CERT division at the SEI. He performs critical infrastructure and protection research, develops tools, methods and techniques for evaluating capabilities and managing risk. Now I'd like to turn it over to Matt. Matt, good morning. All yours.

Matt Butkovic: Good morning, Shane. Thank you. Good morning, Art.

Art Manion: Morning, Matt.

Matt Butkovic: Thanks for making the time this morning to discuss the SolarWinds incident. So I was hoping, Art, you could give us a brief overview of the software issues leading to the SolarWinds incident. I know that maybe calling it SolarWinds without explanation isn't doing it justice, so over to you to maybe give us some background on the situation.

Art Manion: Sure, yeah. Thanks, Matt. Right, like any story that came out, SolarWinds was the word, right, the vendor's name this was first associated with in December. But I think you're right. Evidence is that this is a much larger set of incident activity than just the SolarWinds software. Lots and lots of issues and I hope to get to some of them today during this webinar. Lots of software-related issues that led into this, and I think this activity serves as a reminder of some things.
And perhaps the first, it might be a bit more of a meta issue or a threat issue than just a technical software issue. But please let this be a reminder that adversaries do target software build, software development, software deployment systems. The SolarWinds incident activity here is just one of many such attacks that are publicly known about. But there are big questions about how you trust and assure the software is the software, hasn't been modified, and this really draws attention to the threat of people are actually going after this class of target.

My team was brought in. A lot of our day job activity is coordinated vulnerability disclosure. So we were brought in by DHS CISA, one of our sponsors and partners, to look at what was effectively at the time a zero day or a suspected zero day vulnerability in SolarWinds. And an early question back in December was, was this vulnerability being exploited as part of this

incident activity? The results of that line of investigation sort of led to yes, that vulnerability was being exploited, and malware was being installed on SolarWinds installations. But that seems to be independent activity from the supply chain build system compromise activity.

So early on, we're looking at a vulnerability. Hours, or a day later, the phone calls with CISA and SolarWinds turned to compromised build system and the incident began to unfold. And again, in those early sort of days and hours, we weren't sure what was going on, but again, my team was looking at the vulnerability aspect of this. In the end, there's a vulnerability note in a CVE entry, common vulnerabilities and exposures. So the vulnerability in question is documented and published and fixed. SolarWinds has fixed a number of other vulnerabilities, but again, there seem to be two threads of activity here. This one vulnerability and a .net web shell malware being installed, again seemingly independent from the supply chain attack.

Matt Butkovic: I'm sorry. I know there's a lot to parse in this conversation. It's a very complicated topic. Just a follow on question if I might, which is for our audience. I've heard references to Sunburst and Supernova as two separate vulnerabilities. From a practical perspective, how much does that matter in our ability to understand this incident?

Art Manion: Well, you know, identification, naming of vulnerabilities, naming of adversaries, naming identification of malware is a giant mess. In fact, early on, I confused a number of these names. So, you know, in a sense it matters, if you are doing technical-- if you're documenting this technically, getting the names right or figuring out the relationships of the identifiers does matter. I think my bigger picture take is specific to the SolarWinds part of this activity, there still are these two threads. One which is the vulnerability I talked about. That one's called Supernova, and SolarWinds calls the vulnerability Supernova, and this web shell Supernova. The other names are for the supply chain activity.

Matt Butkovic: Thanks, Art. Just to boil that down, it sounds like the technical remediation activities are in essence the same. We'll apply some specific software fix, but your distinction is really important, which is, you have something that looks fairly standard, which is a vulnerability detected in software. The more interesting story is perhaps how that vulnerability was introduced into the software supply chain. So I was hoping, Art, that this morning, you could explain to the audience how this situation with SolarWinds is different than sort of the standard or typical software vulnerability discovery and exploit life cycle.

Art Manion: So again, the thread my team was brought in on is, Matt, in your words, the sort of more standard vulnerability exploit life cycle. There was an inadvertent vulnerability developed by SolarWinds in their software. Some adversary discovered this, developed an exploit for it and was using that to break in to SolarWinds installations.

But again, seemingly independent activity, so perhaps a different adversary, compromised SolarWinds themselves, their infrastructure, their dev infrastructure, put in very elaborate malware, then very, very surgically and carefully modified SolarWinds' software build processes and put a Trojan horse or compromised SolarWinds' updates. So the bigger-- the build process compromise activity is much different than, okay, attacker found a vulnerability. Attacker exploits it. That gets noticed. Vendor patches. We go onto the next one of 20,000 or more per year public vulnerabilities.

The supply chain stuff was much more insidious, very surgical. The SolarWinds timeline is nine months or longer of adversary time on target, with compromised updates and compromised software. So again, that build system compromise is a much bigger story, much more elaborate activity than a relatively straightforward, attacker finds zero day. Attacker exploits zero day. Vendor notices zero day. Vendor fixes it. Everybody patches and repeat that cycle.

Matt Butkovic: Yeah, thanks. I really took a note. You were describing in sort of the introduction to the incident that the adversary, or adversaries, will target the development pipeline. They will specifically look to exploit this life cycle, and I think that's really a key point here, which is, this is pernicious. This wasn't a misunderstood technical flaw that then found an exploit developer. This was the specific targeting insertion in the software supply chain that allowed for exploitation and I think that that nuance is certainly not lost on folk. So I think about the popular media depictions of this incident, but I think it probably shows the way to future incidents. And I know we'll talk about trust a good deal this morning, but it seems to me that the thing that rattled me about this was, this incident very specifically laid lie to the fact that there are trusted components in your supply chain that are unevaluated.

Art Manion: Yeah, the trust, in other terms, sort of integrity of your supply chain, software assurance, confidence that the software has integrity, hasn't been modified. Here we are in 2021. Software systems are everywhere. We depend on them. They mostly work well and benefit us. But we are all extending a great degree of trust in the fact that the supply chain is uncompromised, and here's a clear example where it was compromised, unknown to many people for nine to 12 months, and this is the one we've noticed. We've noticed others in the past. ASUS Live, the NotPetya malware, was probably delivered via compromised Ukrainian accounting software. There are many examples of this. There's a great reference back to trusting trust, Ken Thompson, 1984?

Matt Butkovic: I believe so.

Art Manion: Yeah. We're all kind of trusting a lot of stuff, and that trust may or may not be well placed. It's a tough one, yeah.

Matt Butkovic: I love making that reference to Ken Thompson's seminal paper, because I think it demonstrates that some of those foundational concerns that Ken identified in 1984 are still with us. They're just sort of changing form, or moreover, becoming more complex, right? In that paper, he flipped forward, and if the audience isn't familiar, definitely check it out. It's, I believe, an IEEE paper, and it asks the question, how do you trust basically the compilers' functioning, if I recall correctly? These questions of trust will always be with us, so one of the points I'd like to make and I'd like your thoughts on this Art, is that the technology stack will change. It will evolve over time. But some of these concerns are enduring and there isn't an easy, simple fix.

So understanding the things that are trustworthy, in my mind, comes back to the idea of verifiable confidence. How do we know we should trust a component? Or in this case, trust a vendor? One of the things that I draw attention to is, most organizations don't have the capability to independently evaluate software in the way that we might at CERT, right? So how does your organization establish justified confidence in a vendor and its products? Not an easy topic.

So Art, I want to just pause there. There's been a number of audience questions. Are you okay taking those now?

Art Manion: Oh sure. Yeah, yeah, absolutely.

Matt Butkovic: Okay, there's really good response here in the livestream. The question, or the ask of us is-- and really to you, Art-- can you compare and contrast SolarWinds to what occurred with the Apple iOS a few years ago? So let me stop you there. Are you willing to talk about that topic? Is that something we can do?

Art Manion: I would be willing, but there are a lot of-- I don't know specifically which Apple iOS thing. Sorry. There's a lot of them, so yeah.

Matt Butkovic: Yeah. I had the same thought. So Richard, if you wouldn't mind elaborating on what exactly you're looking for us to address, happy to do so. The next question: what can organizations do to detect and protect against this type of attack? Now I think that builds directly on the discussion we're just having. So from a technical perspective, you are a leading expert in vulnerability identification and analysis. What can an organization do? And I would say, what can the average organization do to identify this sort of attack?

Art Manion: Yeah, I don't want to sound overly pessimistic here. We mentioned this up front, right? By all accounts and by all available evidence, the threat actor here is a well-resourced, well-funded, skilled, patient nation state type of adversary. It is, quite honestly, very difficult for anyone, much less map to an average organization to defend against that class of attack. And I'm

not making a judgment about anyone's following the right number of those controls or the wrong number of controls, or which things will work and which things will not. But just keep in mind, this is often called the defender's dilemma, right? My organization, I have to protect every possible entry point, every way in. The adversary needs one functional way in or a couple of ways in. So the defender's in perhaps a not great position to start out with.

Nonetheless, there of course are things we can do. When I first realized, back in December, we were looking at the vulnerability. It became a bigger incident and I was thinking, oh, did the attacker compromise the SolarWinds source code? Did the attacker steal their code signing key? And in fact, apparently those things did not happen. So in this case, the code repository was secure, and not tampered with. The code signing key was secure. This attacker had very, very carefully-- they were monitoring the build process, and as the software was being built, they replaced one file and then covered up their tracks and sort of stepped out of the way again.

So I was looking for a standard-ish, average mistake, right? Forgot to code sign. Didn't protect the code signing key. Unfettered access to the source code repository. Public information and evidence suggests that's not the case. So we have some standard practices. They were apparently followed. They didn't help in this case.

Again, I would implore development organizations to really perhaps reevaluate your threat model, or the risk assessment for your build and development and deployment processes. They are targets. Please realize that. Please take really, really extra care to shore up the integrity and the authenticity of your software.

User side, I'm a downstream customer or user of the software that got compromised, again, it's a little bit tricky. I could perhaps run updates or test software in a sandbox environment. However, it's been common practice for years or decades for malware to look, to see if it's being run in a sandbox, and not active. Again, we get back to the trust, right Matt? I'm going to take a leap at some point and trust that the software is from the vendor, it has integrity and the signature checked out. There are not, in my mind, a lot of great things to do. Nothing leaps out to me as a clear miss, as a clear step someone should have taken to just block this class of attack. We have a really highly skilled adversary here.

Matt Butkovic: Yeah, so Art, I'd love to build on that, if I may.

Art Manion: Please.

Matt Butkovic: I would also put the plug in for operation resilience. So you're not going to be able to effectively defend against all of these attacks. That's reality. It doesn't mean you shouldn't try. It doesn't mean we shouldn't make it more expensive for the adversary and run them out of money and energy. However, disruptive things are going to happen to organizations.

So I want to get back into the technical details of SolarWinds. But I'd like to offer this to the audience: knowing that you will have some disruptive event in your organization, all the more reason to pre-position incident response resources and the tools needed to find those indicators of compromise and then recovery.

So if you think about having an obligation to both protect and sustain, you've got to strike a balance, right? Your protection's going to fail at some point, and you're going to invoke these other capabilities… Determining what those are at the time of need is a really bad idea, right? You need to know what those are, the quantity and quality of your response capabilities, and then test those, in advance of something like the next SolarWinds.

So Art, you said something that I thought was really interesting, which is, SolarWinds also defeated some of the very safeguards we thought were most important to thwarting this action. Software signing is a great example (inaudible). At some level, you have to put trust in your vendor. Again, it's not a binary. It's not trust or not trust, but I would suggest we need some sliding scale, based on evidence, about the trustworthiness about our partners and our software. I'll let you reflect on that, Art.

There's many questions. We're tracking lots of questions. I might just press on with those, if you have any capping thoughts for my comments about the mix of capabilities in an organization.

Art Manion: No, if you don't mind, Matt, I do see one-- I noticed one here from the audience I wouldn't mind taking a crack at.

Matt Butkovic: Of course.

Art Manion: I believe the person is asking, would earlier detection of the compromise have helped? I believe that's the nature of the question here. And of course, yes. Again, we have a pretty stealthy, pretty skilled attacker here. Of course, there are techniques to look for beaconing command and control channels. So SolarWinds was compromised. There was malware on their infrastructure. Could SolarWinds have detected that sooner? Maybe, arguably. It's very hard to make that judgment, not being there in person or having inside access to that. So possibly. I'm not an IDS detection person. I'm not strong enough in that area to really make a judgment there.

And one quick thing, Matt, to go back technically. I read a bit about the Visual Studio sort of build environment. SolarWinds software is Windows Enterprise software, it's .net. There is a sort of a reproducible build integrity mechanism that a program database file, the PDB file, is produced when Visual Studio builds software. That file contains a cryptographic hash of the source code files it used to compiles the software. So presumably, checking those hashes may have indicated that what we have in our source repository and what was used during the build were different hashes, and therefore different files. That's a pretty detailed technique that may

have shed light on this incident activity earlier. It's my layperson's understanding that if the attacker has control over that build infrastructure, they could perhaps just lie and modify this PDB file as well, and defeat that technique also.

But depending on your build environment, there may be cryptographic signing mechanisms, double, triple checking mechanisms, to stop this sort of attack. I don't know what those are exactly are. Development is not my day job. And that will be environment-dependent. But again, with the appeal for developers to be very, very sure about your build process, all of the extra integrity checking you can turn on, now's a good time to consider that.

Matt Butkovic: Excellent suggestion, Art. I just want to make sure we're setting the right tone, which is, I don't think you and I are advocating that you should omit or somehow diminish the importance of any specific controls, but rather, I think we're pointing out the limitations of some controls. In that spirit, there's a question from the audience I'd like to explore, which is, the question is: should this be or could this be a turning point for zero trust, the zero trust approach? I think it's a really interesting question. Art, would you like to take a shot first at that? I've got some thoughts as well.

Art Manion: Sure, I'll take a partial shot and this may be a bit of a side angle on this. Recently, I think it was just last week, SolarWinds published a blog post, sort of getting into how they think they were initially compromised, and they talked about a user account compromise. And on the other end of this activity, what happens if I download compromised SolarWinds software and I get that malware and it beacons out and I'm selected? I'm one of the lucky 18,000 customers infected. I'm one of the lucky winners who gets additional attacker interest. That activity is also often off indication authorization identity, Windows active directory, hybrid active directory, Azure cloud active directory. So I think the question really is, is (inaudible) identity and zero trust going to help here? It could. I'm familiar with sort of the BeyondCorp and the zero trust ideas. That model seems to be decent for a lot of reasons. Google is probably living proof that it can work. But even then, the identity databases, the authentication databases, that's where attackers are going.

Matt Butkovic: Right.

Art Manion: So you still have centralized risk and somebody pops that-- gets into that centralized identity database, which is where attackers go, once they get on your network, and how they may be (inaudible) in the first place, ultimately, you still have to defend at some point. So I love the zero trust ideas. I think they're technically interesting and probably useful. They're also not a silver bullet.

Matt Butkovic: I would agree, Art. It's my opinions as well. As a set of concepts, I think it's a really good idea, or set of ideas, right? We should be pursuing solutions exactly like this. Will

this hasten the adoption of zero trust? It may. Will it fundamentally fix the issues that led to the SolarWinds compromise? No. That doesn't mean it's not a good idea.

I'd also add that there's a question here about the potential for a malicious insider being at the heart of this. So we certainly know there was an adversary that was well-resourced and patient, and well-organized. Again, because we haven't said there's a malicious insider doesn't mean an attack like this can't come from an insider source. I don't think in this case it was, based on what we know. Also, I think coupled to this question is an idea about fourth party, right? So there's some suggestion, Art, and I'm not sure how much you've had an opportunity to research this, is that SolarWinds may have been drawing on other software development resources external to the organization, so as we call, the fourth party. The question then becomes, what visibility did SolarWinds have to that portion of their supply chain? And by proxy then, an extension of your sources of risk. So again, I don't think there's any easy answers here.

Something I want to draw attention to, and I used the phrase silver bullet. I think most of us came to this profession or through this profession as technologists, so we have great faith in the next technological leap is going to correct something or make it better, which is true. But SolarWinds also to me shows the limits of that approach. We're talking about process problems, coupled with technology problems. You could argue, as we've done for many years at the SEI, that all software defects are the result of flaw in process, but at a very practical level, something I find a bit off-putting is that we're awash now in vendors saying that there's a magic box with blinky lights that will prevent the next SolarWinds. Be smart consumers of these things, right? There is no magic fix for this, but rather a set of interlocking technical and process changes that can help prevent this. But if someone says their appliance will prevent SolarWinds, they're lying to you, and be careful about snake oil, is what I would say, Art. Maybe that's too fine a point, but would you agree that that's something we have to watch for as practitioners?

Art Manion: Certainly. There are any number of jokes about the conference showroom floor, blinky lights, magic boxes that do things. I think it was maybe a Rob Joyce-authored blog post from a while back. I think he was still at NSA at the time and the flavor of it was, in his offensive role, what he did not like to find was a target that was doing the basics wrong, because that made it harder for offense. My general advice is, if you're not doing the basics, you don't need the magic blinky box. Start with your basics and core auth, core authorization, authentication, good identity management, sign-in code, sign-in code check-ins. There's a bunch of practices you can follow that make this harder. But again, we covered this already a couple times. The adversary does have some innate advantage and it's really hard to just block all of these things.

Matt Butkovic: That's your problem, right? We'll have this kind of asymmetric warfare perpetually. I do have a question, Art, before this slips away, if I may. So there's a number of audience questions about the QA process and the vendor's QA process, right? So basically, how much faith should we put in a vendor's QA process? I want to ask you that, since you're really

the expert to ask, but I want to put a slight modifier on that, which is, what is a reasonable level of transparency? What is the expectation that a customer of SolarWinds or any other software should expect to see in that software development pipeline? How much visibility and confirmation of good practice should we expect as consumers of software?

Art Manion: Well, I'm a strong proponent of more, or as much-- short of giving out intellectual property crown jewels, trade secrets. I'm not here asking a commercial proprietary code organization to publish their source code, but as much transparency as an organization is able to provide, I think that's going to help a lot as things are developing and towards the future, as customers and users become more aware of these class of problems. This is a trust but verified, it's the verified part, and I can't verify it without some transparency from my software developer, from my provider, from my supplier organization.

So a document describing-- back to your point, processes, right? We follow these processes when we build code. Don't give us your secret code signing key, but say that you have a key, that you treat the key carefully, that it's on a separate independent system. Access is controlled. Describe some of that stuff. You talk about the fourth party. In this case, the detection, sort of on-paper detection is easy. If I have SolarWinds and I updated in the last nine months, I probably got compromised. But what was upstream of SolarWinds? Some other product? What about the fifth, sixth, seventh, eighth and thirty-seventh parties upstream?

Matt Butkovic: Right, absolutely right.

Art Manion: So I'd also implore software developers to be very careful about their upstream dependencies and their supply chain. At the very minimum, know what you've got and have a way to find out when your upstream suppliers are fixing vulnerabilities, because now that's something you have to take care of as well.

Matt Butkovic: Sorry. I'd just like to expand on something you mentioned, which is, having this multitude of additional parties. So I'm going to step back from the technical and approach it from a supply chain risk management perspective for just a minute here. So you're going to need to build trust in these third parties at arm's length. That's the nature of having a third party, right? The cloud relationships we have are great examples of this. Your SLAs will determine what you're entitled to know, right? Couple that with, you should probably receive-- undoubtedly you should receive, but you often may not-- a robust understanding of the control environment operated by the third party. This is where things like SSAE 18 reports are really important. The SSAE 18 requires that the service provider enumerate and then evaluate everyone in that chain that leads to your service. So that's a separate webinar. But I would just say, it isn't just code signing. It isn't just the tools that we have at a technical level. I would argue the process parts of this and the strategy employed are equally important. So I don't want to get stuck there, but I just

want to mention. I think there's a number of process tools and leverage capabilities outside of the technical realm that can help you get your mind around this risk.

Art, you've been a proponent for years of the software build materials, right? And I know you've advocated-- and I'm a believer-- so I'd like to make some more acolytes today. Can you describe the software build materials and how it fits in some of the solutions we're suggesting today?

Art Manion: Yep, sure. We just covered it without saying it, right? The parties beyond yourself, second through nth upstream, right? Upstream dependency, upstream dependency. Oftentimes this takes a flavor of some proprietary or commercial software vendor, provider, incorporates an open source library from upstream. There often is discussion about incorporating open source software in proprietary software. That's not the only form this takes, but that's pretty common.

What we're really getting at is, it's very hard to defend your software if you don't know what software is in your software. It sounds dumb when I say it, I think, sometimes, but it's that simple. And 2021, super complicated software, deep supply chains, complicated supply chains. People ask my team, "Hey, who all is affected by the vulnerability in that library?" We take the best educated guess with 20 years, 20 plus years, of multiple people's experience, and keeping notes from the last time we handled a case. That is not a scalable good data-driven solution to knowing what software is in our software systems.

So S-bomb is very basically, the bomb is built of materials, right? Entire physical industries only work because builds of materials are part of the process by which you put a component in an automobile, for instance. We are advocating that a software version of that is sort of the least cost avoider way, right? A vendor builds software. Their development tools can only build the software if they know what's in the software. A byproduct of build and development is, you know what you just put in there.

Start making that transparently available to your downstream, your user, your customer. And then if it's turtles all the way down and a big network effect, if every vendor does this, every supplier does this, and we reach the critical mass, all this transparency starts to be visible, and at any point in that giant supply chain mess, someone can look and say, "Oh, I used this software, which uses this software, which uses an open SSL version, vulnerable to Heartbleed from back in 2014." And that knowledge today is lacking. We are doing guesswork. We are doing binary software composition analysis and source analysis, which works to some degree, but it's an expensive after-the-fact investigation. S-bomb, tell us up front what's in the software.

Matt Butkovic: Art, you know I'm a believer. To me, to draw a strong analogy with the physical world, a product recall, right? How do you know that the specific item contains melamine? Well, without a list of materials that go into the product, you can't do that.

Art Manion: Right.

Matt Butkovic: And I think that's an example where it's really customer-driven, isn't it? So if the customer were demanding this. As you said, if there's critical mass to require vendors to do this, it'd put us in a better place. Again, it wouldn't have prevented SolarWinds.

Art Manion: Right.

Matt Butkovic: But a stronger software build materials may have helped you unwind the SolarWinds situation quicker. Is that fair?

Art Manion: No, that's great. And actually, thank you, Matt, good point. I got excited about S-bomb and forgot to mention, I by no means want to indicate that S-bomb would have prevented or stopped this SolarWinds attack. Yes, it may have helped. Better supply chain hygiene may have assisted in some way, but in this case, right, I know I had SolarWinds software. I didn't need S-bomb transparency to tell me five hops away was SolarWinds. I have SolarWinds on my computer. I know that it's there. So again, what's in my software transparency wasn't going to help here, but making knowledge and first order consideration of what's in my supply chain a regular thing, a part of your risk assessment, a part of your threat analysis, having fewer, better suppliers, that's where there may be some sort of secondary benefit from a more S-bomb enabled world.

Matt Butkovic: Yeah, absolutely, Art. I think that the transparency of S-bomb allows us to make better risk-informed decisions, right? And that's what we can hope for. That's the best consideration. There's a question here about incident response.

Art Manion: Yep.

Matt Butkovic: I said that sort of broadly, and maybe didn't elaborate, but what I'm suggesting is, you need to have an appropriate level of capability to identify, triage, respond and recover, right? You need at a minimum to have those things. And that's going to invoke a number of related process areas. That's something that's at the heart of our work here at Software Engineering Institute and CERT, which is understanding that these aren't disparate topics, but rather an ecosystem of capabilities. So we can point you in the direction of more specific guidance regarding incident response.

Art Manion: Yeah. The body, volumes of work on what incident response management should look like, partly written right here at the SEI over the years. You need some capability, but the details are somewhat subjective to your ecosystem, your environment, your organization.

Matt, I'll just quickly, on the IR topic, a bit of perhaps out of order advice. In the December timeframe, SolarWinds was a little bit stuck in a position where, I mentioned in the beginning, this vulnerability they wanted their users to update, apply a patch or an update to remediate or fix a vulnerability. At the same time, their downloads for their updates for their software were still compromised. So there was some advice about, go get this patch and update to protect yourself against the vulnerability. But I would argue that there was significant lack of trust at that moment in time in the integrity of the updates. That was a pretty ugly catch 22. And then furthermore, just patch your SolarWinds is not the first step in that response.

Matt Butkovic: Right.

Art Manion: The adversary was on target for nine months. I'm sorry, you have a very expensive investigation to do, and you have to make an organizational decision. Was I in the 18,000 customer set that got more attacker interest, or was I in the 18,000 set that malware pinged home and took no further action? You need to figure that out. And depending on your decision, on your answer there, we're talking white boxes rebuild. Rebuild from a clean, known, good software. And way at the end is update your SolarWinds software. So order really matters in incident response like that. There's a CISA document out that puts it in nice order. But some of the initial SolarWinds advice was focused more on the patch and less on, make sure you've recovered from the incident first.

Matt Butkovic: Thanks, Art, and it's a great point. I think the guidance that, turn it off if you're affected, is a less than-- I mean, it might be absolutely spot on, but I don't know how practical it is in some settings, right?

Art Manion: Yeah.

Matt Butkovic: From a triage perspective, if you think it's risen to a level of risk where turning it off is the safest approach, totally understand, but I think for many of us, that's a very difficult option to (inaudible).

Art Manion: You're right on, and it depends a lot on the environment, you know. Here this is SolarWinds, so you know, hand-waving, network monitoring and management software. Maybe you can run without SolarWinds. Control system software, running a production facility, running energy distribution, those things we don't want to have turned off. There are huge physical world impacts there. So unplugging and turning off in that situation is a much different discussion than, okay, I'll live without SNMP and network management for a couple weeks.

Matt Butkovic: Yeah. I think that's a great overlay, which is in a sense, the SolarWinds incident is hugely disruptive, consequential. If we have a thought experiment where it's not a SolarWinds package, but rather a safety-critical software package, running on production line

controllers, on something that is immediately impactful in the physical world, I think we leave really scary dilemmas regarding turn it off, right?

Art Manion: I believe those will be much harder decisions than turn off my network management stuff, yeah.

Matt Butkovic: And I should point out that we've seen versions in the past of SolarWinds-like attacks in exactly those sectors, right? I'm thinking of the Havex and Dragonfly campaigns from seven years ago. So I think one of the things that folks maybe don't fully appreciate if they're casual, or casually looking at this or new to the topic is, that the SolarWinds attack is maybe a watershed moment because of the scale a complexity. But this attack path has been tried before and it has been successful, just in a more limited way.

Art Manion: Yeah. We named a couple of things earlier. The person who has been asking about Apple iOS added a bit of context there. Thank you. Right, Apple, iPhone, iOS app development software was basically back door or Trojan to some extent, and that spread through the store ecosystems. Yeah, at one level, these development-oriented build system attacks are sort of all the same class of thing. They have happened, they will happen. Some of them were very hard to root out. If you have to go recompile every app that was built with a compromised module. There's some funny-- well, I can say funny because they didn't affect me-- but in the modern, superfast JavaScript-based web development world, there's the Left Pad incident, with no JS. In that case, there wasn't malicious activity. It's just that the software went away, downstream, breaking a whole bunch of websites. And that was just-- that wasn't an insidious compromise, that was simply a denial of service more or less.

So again, please leave this webinar remembering that development build system attacks are a huge thing and, you know, consider that very highly in your threat models. And Matt, really quickly, there were a couple of insider-- you mentioned insider threat earlier.

Matt Butkovic: Yeah.

Art Manion: I agree with you, Matt. I've seen no evidence in my public sources of that here. But right, if I put myself in the attacker-- put my attacker hat on, and I'm a well-funded, state-organized, sponsored adversary and if I'm going to spend, I don't know, I'm going to pick a random number, a million dollars on attacking, on this attack. If I can just pay someone 500,000 and they'll give me the sign-in key, you know, that works too. So an insider, of course, is an avenue here. The SEI's got a big body of work on insider threat. Not my area. Matt, you know more about this certainly than I do, but sure. People are a path, absolutely.

Matt Butkovic: Yeah, absolutely. It's just one path, right, among many. In some ways, it shortcuts the hard work of other paths. So a webinar for another day, Art.

Art Manion: Yes, absolutely.

Matt Butkovic: There's a number of questions or comments here regarding response and I would argue that yes, a TTX, a tabletop exercise that games at this is absolutely a very good suggestion. And also, your business continuity and disaster recovery plan should account for this, right? One of the failings I see is that business continuity and disaster recovery, oftentimes the plans speak in a too-generalized way about some sort of cyber disruption. I think SolarWinds, at a minimum, should serve as an input to say, we should be conducting tabletop exercises, perhaps technical exercises, and updating our plans to account for this, in the same way that I watched ransomware change disaster recovery plans and business continuity plans over the last three or four years.

Art Manion: Yeah. Again, no defense will ever be perfect. There will be incidents. We need emergency response. You need incident response. Please, please do a dry run first, and figure out what's important and what's necessary for your organization. It's not entirely one size fits all, but this is the modern world we live in. You're going to need to respond to attacks.

Matt Butkovic: So Art, a question occurs to me. We've been talking mainly about the software consumer side, right? You've made some description of things that we could do in the development life cycle, or supply chain. Other thoughts about what developers specifically can do to ensure that they're doing their best to avoid this sort of compromise?

Art Manion: You know, honestly I don't-- partly beyond my strong area of experience, for sure. But whatever development ecosystem you work in, right, and SolarWinds was .net and Visual Studio, which caused me to go spend a couple hours reading about the program database files I mentioned earlier. Understand all of the integrity protections that exist in your development environment, and you know, at least consider, if not perform, all of them. And again, your code is a target. It may not be the target today. You may be a bigger target because you have a lot of reach into a lot of Windows enterprises, or your software has privileged access to a lot of the systems it runs on. You may be a bigger target or a smaller target. But you may just be a passthrough. If the adversary is after something three targets away and compromising your software development process, it's just one of the steps the attacker is going to take, you still might be a useful target. So yeah, please, please, again, turn up the knob on integrity protections for your software builds, please.

Matt Butkovic: Thank you, and you've said this once already but I think it bears repeating which is, ensure you're protecting the software repositories, right? I think there was somewhat of a complacency that settled in around that, right? The kind of ubiquity of code, and the availability of code, that maybe we're not considering-- at least in my opinion. Maybe you disagree, Art-- that we're maybe not pulling on that risk thread hard enough these days. So I'd

suggest, think hard about where your code-- I mean, there's so many examples and incidents we could cite where a lack of adequate safeguards over the code repository leads to something really bad downstream, right? And that lesson should not be forgotten. In fact, it should be reinforced in all of the ways we model these threats.

Art Manion: Yeah, I mean, I'm not sure I have a whole lot more to add, Matt, but you're spot on. This is a thing we've been living with, we're going to have to continue to live with. Please, please, please, turn up your focus on this class of attack.

Matt Butkovic: Yeah. So switching gears slightly, Art, there's a number of questions or comments about SLAs, service level agreements, and then the cloud. I would say that, right, the cloud, because you're going to be managing the situation at arm's length, because SLAs and the contract are going to be your only vantage into those operations done at arm's length, all the more reason to understand in detail.

For instance, maybe it isn't front of mind for security practitioners that have a more technical bent, but it's imperative that you understand SOC reports, right, if you're going to put faith in a vendor. So again, this is another webinar in the making, but if fundamentally you can't discern the difference between a SOC 2 and a SOC 3, or a SOC 2 Type 2, and a SOC 2 Type 1, this is something that requires, I believe, a little investigation. Because you're substituting those assessments of-- third party opinion assessments of the vendor, for your own ability to interact with the control environment of that vendor. You don't have the ability to directly audit, therefore you must depend on others.

Art Manion: Right. Yeah. I've got nothing really to add there, Matt. I just guess I'll say, there was a couple of the questions seemed to have sort of the S-bomb and cloud discussion or topic brought up. I'm pretty heavily involved. I gave a link out. Hopefully that made it out to YouTube, but the Department of Commerce NTIA has a long running S-bomb working group project going on. The developing position there is that if I'm an end user, cloud customer, that cloud provider telling me all the things they use on a regular basis, possibly multiple times a day, probably not super useful to me in my (inaudible) matter, my risk assessment, right? Now that cloud provider knowing what they're using, very important for them to be secure. And again, in a case where the actual ingredients are changing so quickly, the cloud customer may not care. The cloud provider certainly should. And Matt, I think, you know, processes followed by the cloud provider become a much stronger metric than something else you might be able to count.

And again, if it's a third party who has to do this because, as the customer, I can't see inside the cloud, I don't see another model. And actually, I would probably defer to you on this. Not my strongest suit, yes.

Matt Butkovic: A couple of thoughts there, Art, thanks. There's a comment here that I think is worth exploring. So the comment is, "Can opting for cloud software as a service solution transfer the risk to some extent?" I think the answer is no, and I'll explain what I mean. There's a false sense of risk transference, right? When you outsource or draw on a cloud provider, the negative outcome or the consequences if something goes wrong, that is going to fall primarily on the service consumer, rather than the provider. So certainly, the cloud provider will have reputational damage. They may owe you something in the way of restitution. But if it's your IP that's stolen, if it's your key applications that are disrupted, that's going to exceed the exposure that the provider has, in most cases. So I guess what I'm really saying in short is, don't be lulled into a false sense of safety because it belongs to a cloud provider or an outsource, right?

I think in many ways-- Art, I'm thinking back to when I joined the SEI from private industry, about a decade ago. This was early days of the cloud, and there was almost a willful or gleeful ignorance about what the cloud provider was doing. The idea was, if we've outsourced it, if we've sent it to the cloud, it's no longer our problem, and I think that's come back to haunt many organizations. And I don't think the fix for SolarWinds is, let someone else run SolarWinds for me, because ultimately, that SolarWinds environment is going to touch things that are important to me. So all the more reason to understand how it's being operated.

Art Manion: Yeah, I think Matt, you know, the old, or new adage, right, that cloud is somebody else's computer, you're getting right on that. Yeah, sure, I turn my costs into rental as opposed to ownership, or I transfer some risk or some operational methodology with cloud, absolutely. But again, it's not a full risk transfer, probably, in most cases. Actually, technically, I'm struggling. Someone had a comment about this. It may entirely be possible to run sort of SolarWinds in the cloud, but I actually don't know what that looks like, and from my previous brief career as a network sort of administrator, allowing a third party via the cloud to touch all my stuff sounds like a no. I don't think I would take that position going on. Maybe these days, that's super effective, very productive cost-saving network management. But even a dumb home device, where I get a home router and it says, "create an account on our cloud service with your phone," and that's how you manage the device. Oh, that gives me the risk heebie-jeebies, is the technical term, Matt.

Matt Butkovic: Yes, and there's a scale for that, right? The risk heebie-jeebies. Yes, I agree, Art, which is, remember, there's always a trade out to be had. So if you're going to centralize administration of devices, you're then concentrating risk in a way. Again, this is not Art and Matt telling you not to do these things, but rather, I would say, Art, if I may speak for us, imploring you to think about the risk implications of those decisions, including SolarWinds, as the central example to this.

Art Manion: Yeah, please. Thank you for reminding me. My assumption here, and I can easily understand this conceptually, right, having SolarWinds-- having enterprise network management

software, enterprise backup, enterprise antivirus, anti-malware, right, my assumption here is that the costs of operating that stuff and the risks that come with it are outweighed by the benefits, right? In a sense, we wouldn't be running them otherwise.

But keep in mind, you know, Matt, you nailed it, right. Centralizing risk, I've got enterprise agents running with full privileges on my Windows clients. I've got a centralized database. I have domain trust going on. I have domain add-in credentials accessing all my boxes to do antivirus, to do backup, to do network management. There's a lot of exposure with those systems, technically, and a lot of centralized risk.

So it's not my place to assess for a certain organization, do the benefits outweigh the risks of installing this class of software? Back in December, we were investigating this vulnerability. A colleague and I were downloading SolarWinds installers and running virtual machines and trying to figure out what was going on. SolarWinds installers are gigabytes, 45,000 files, 15 gig of installed files. And with no further knowledge of what the software is, SolarWinds' or someone else's, that is automatically exposure, just on a statistical level. That much software has got to have bugs in it, has got to have security exposure. Domain admin access, things running in system on all your boxes, might be worth it. A lot of risk bundled in there.

Matt Butkovic: Yeah, exactly, and risk needs to come with commensurate reward, right? So understand your tradeoffs. Something, Art, since we have a minute here, I was going to highlight some elements of the SolarWinds response that may not be front of mind for those that are technologists. So Art and I had an opportunity to speak with folks in the insurance industry about SolarWinds. And I can tell you that I believe SolarWinds will be cause for a re-examination of what a good insurable risk is in cyber, right? So there was a question about kind of worst case scenario.

One of the worst case scenarios is there's tremendous loss because we're replacing equipment, or there's other disruption, right? So the way that your insurer looks at you as a risk may change because of SolarWinds and maybe that's a good thing. The way that your board of directors looks at you and the questions they ask you will undoubtedly change, right? I can tell you that you're likely to face new questions from senior leadership, because they're being asked those questions of their insurers. They're being asked those questions of the folks in finance, because this is such a prominent event in cyber security.

Now I'm not suggesting that we sort of manage by crisis, but I think SolarWinds is one of those invents that there's going to be a long wake to this, and that wake is going to include being asked questions from perhaps nontraditional camps about how you're protecting your organization. So Art, what I'm really saying is, one of the things that we need to do is translate these highly technical topics into a message, a narrative, that nontechnical audiences can understand, right? Because ultimately, to make progress, we have to have the same or a comparable understanding

of not only implications for future incidents, but how you combat SolarWinds as it now. Like, let's be clear. This isn't over, right? SolarWinds is still unfolding. So there's fixing today's crisis and then preparing for tomorrow's crisis. So we're closing in on the end of the time that we have, Art. So I wanted to hand it back to you regarding, are there specific resources? Are there SEI or CERT artifacts that you'd point to that could help the audience, thinking about SolarWinds?

Art Manion: I've been building since back in December a sort of a collection. It started out as my personal notes, but a collection of resources and references to SEI and other places, that are both sort of evidence of how we can assess this risk, but also what might be done about it. So we're going to post all that, a loosely curated bibliography. We'll post that in the YouTube channel, perhaps elsewhere.

It's going to depend a lot, but if you're a development organization, I'm going to assume you have developers who know how to operate their environment. Again, go double check all of the integrity protections, all of the processes around code signing, check-in, authenticating who made changes to code, all your build processes. Please, please, please focus on that.

And Matt, what you just said really, I think is spot on again. This is an opportunity. Don't manage by crisis, but this is an opportunity to double check our perhaps prior understanding of all the risks tied up here and re-evaluate that. Maybe it's the same. Maybe it's higher, maybe it's lower. I'm going to suggest it might be higher than we all thought. And at the very least, take the opportunity here to re-evaluate that, and continually do that. And Matt, you're right. I mean, history is-- the regression analysis is simple here. There have been these class of attacks. I promise you, there will be more. There are some we don't even know about. So it is happening, it will happen. Readjust your risk and your threat profile.

Matt Butkovic: Yeah, and Art, I think I want to end of sort of a more optimistic tone. I think we've discussed a lot-- not suggesting your tone wasn't optimistic, but I think you'd agree, that this isn't hopeless, right? I don't want anyone to think that we're just kind of traveling, or trading in gloom and doom. So the past is prolog. We know things like this are going to happen again. The question that I think the audience should be asking is, how do we navigate our way out of the current SolarWinds crisis, and then what are those thematic lessons or specific actionable learnings we should apply as a result of our experience?

And certainly, as always, the SEI has a great deal to say. The CERT division has a great deal to say. Please do visit our websites. There's a number of artifacts. They're all available for free. And I would just encourage, if the audience has specific questions, we'd love to hear from you on an ongoing basis to ensure that we can focus our efforts on the things most beneficial to our stakeholders.

Art Manion: Yeah, thank you, Matt, for cheering it up there at the end. I live in the all software is insecure world and trouble world, so thank you for the glass half full version, thanks.

Matt Butkovic: Well, seldom, Art, am I accused of being an optimist, so I guess (inaudible) working in combination. So I believe we're kind of at the end of our allotted time and I would encourage everyone to consider their role in the long term fix to the things we discussed today, which is ensuring you have verifiable trust in all of you third parties, all of your vendors, and ensuring that you process and technical solutions work in a concerted way for a specific objective.

Shane McGraw: Matt and Art, great discussion today. Thank you very much for sharing your expertise.

Matt Butkovic: Our pleasure.

Art Manion: Thank you, Shane. Thank you, Matt.

Shane McGraw: Yeah, and lastly, we'd like to thank you all for attending today. Upon exiting, please hit the like button below your video window and share the archive if you found value in it. Also, you can subscribe to the SEI YouTube channel by clicking on the SEI seal in the lower right corner of the video window. Lastly, our next livestream will be on March 10th, and our topic will be modeling dev sec ops to reduce the time to deploy an increased resiliency. Registration information is available on the SEI website now, and will be emailed out as well. Any questions from today, please send to info@sei.cmu.edu. Thank you for your time, everyone. Have a great day.

Matt Butkovic: Thank you.