Carnegie Mellon University
Software Engineering Institute

SEI Podcasts

Conversations in Artificial Intelligence, Cybersecurity, and Software Engineering

3 API Security Risks (and How to Protect Against Them)

Featuring McKinley Sconiers-Hasan as Interviewed by Suzanne Miller

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

Suzanne Miller: Welcome to the SEI Podcast Series. My name is Suzanne Miller, and I am a principal researcher in the SEI Software Solutions Division. Today, I am joined by my colleague, McKinley Sconiers-Hasan. We are here to talk about security risks in application programming interfaces, known as APIs, and how they impact zero trust. But first, let me introduce our guest. McKinley is a solutions engineer in the SEI CERT Division who recently authored a blog post and paper on this topic, which we will link to in our transcript. Welcome, McKinley.

McKinley Sconiers-Hasan: Hi. Thank you, Suzanne.

Suzanne: McKinley, let's start by telling our audience a little bit about yourself, what brought you to the SEI, and the work that you do here for those who have not met you before. You have only been at the SEI for less than a year. You have already authored a paper, a blog post, and here you are with a podcast. I think a lot of our listeners would be interested to hear how you came to work here and what you like about the work that you do here.

McKinley: Like you said, I have worked here less than a year, about eight months. I came to work here because I wanted to get into cybersecurity about a year ago. Is it not what my background was in, but I thought it would be a good place for me to be, so I did some learning. Long story short, I ended up here at the SEI. I like working here and I wanted to work here because I think the work we do is really interesting. There are constantly new things to be learning and different projects to be working on, and it never gets boring.

Suzanne: That is exactly why I am here too. Is it really hard to get bored. I am glad that we have you here and that you are doing this work for us. We have done previous podcasts on zero trust and APIs. For our audience members who are new to the topic, can you just do a brief overview of the two concepts in case they have not run into them before?

McKinley: Yes, so application programming interfaces, APIs, are like a connector between applications and other applications or users. They can be as simple as someone accessing all of the items on a webpage that a store sells, or it can be as personal as accessing your bank account and seeing how much money you have in your account. Is it just a way of being able to access, modify, or delete data that an application has access to. Zero trust is a concept related to network security. Traditional network security focuses on the perimeter of the network, things like firewalls, whereas the center of the network, once you are inside of it, is not heavily protected. Zero trust focuses on protecting the interior of the network as much as protecting the perimeter of the network.

Suzanne: Very good. In your paper, which we will link to in our transcript, of course, you noted that because APIs are intentionally designed to be exposed to the public, they increase the attack surface of a network, which I can see how that could happen. How does this potential security issue impact the concept of zero trust within a network?

McKinley: Yes, so because APIs are meant to be exposed to either the public or other applications or other entities, it creates an access point in the network, which can be risky. This affects zero trust because it means that those APIs, for one, have to be designed well. They have to be coded with security in mind. The configurations with the API and the infrastructure surrounding it, like API gateways and things like that, also have to be well configured to make sure that they are protecting the network and implementing zero trust.
Suzanne: Back in the day when I was a young software engineer, APIs were

new, and they were actually not trusted at all. The whole idea of having these interfaces that could access something without it being a direct link from one piece of software to another was considered to be very risky. Over time, this has become standard practice, partly because we do have some ways of protecting the perimeter that we did not have back in the ‘80s. We did not even have necessarily what we would call a perimeter back in the ‘80s, but also because they have become so useful as a concept. This means that we have a lot of people that are possibly building APIs that may not have as much knowledge about what the security risks really are with APIs. You outline three top API security risks in your white paper and some recommendations for mitigating them. That is what we are here to talk through today. Can you go ahead and walk us through those, please?

McKinley: Yes, so the security risks are . . . The first one is third-party software integrations. Nowadays, because there is so much software in existence, it is really easy to use other software libraries or pre-made code to create your own APIs. It makes the process simpler, but if those pieces of code, those third-party software integrations, have vulnerabilities in them, then you are potentially putting vulnerabilities into your own API. That can be risky. Then there is also cascading failures. When you create a microservice architecture, which is typically the architecture that APIs are created with, which have a lot of API endpoints, they are supposed to be designed to be loosely coupled, which means they are not supposed to be heavily dependent on each other. As in, if one API goes down, it should not affect all of the other ones. But if they are not loosely coupled, and they are heavily dependent on each other, it means that if one API goes down, it could potentially cause slowdowns or even an entire outage for the network. That is a risk of APIs, specifically when it comes to microservice architectures. Then there is also the increased attack surface, which we talked about previously. Older designs for applications were more monolithic than they are nowadays. Those monolithic architectures tended to have fewer APIs than modern-day architectures do. Because of that, more APIs mean more access points in the network, which increases the risk of potential vulnerabilities.

Suzanne: Organizations need to protect themselves against these risks and take mitigation actions. Let’s start with the third one and move backwards. What kinds of things should an organization do to protect against, as our third one, the increased attack surface? Then we will go to the others.

McKinley: To mitigate the risks of an increased attack surface, one thing you could do is to be strategic about adding more APIs. Each API does create another entrance into that application and potentially into the network. So, making sure that that API is necessary, and the risk is worth adding it to the existing infrastructure. Also, making sure that all APIs are documented, preferably through auto-generated documentation. It is a very common issue that most organizations have, which is having lost APIs, as in they have written them, and then they forget about them. Then those APIs are no longer being updated for potential vulnerabilities, and that creates more risks for the network and for the organization. Making sure you do all those things can decrease the risks of the increased attack surface.

Suzanne: Having a catalog of your APIs is something an organization can do to make sure they keep their APIs updated and understand what their actual risk is.

McKinley: Yes, exactly. You have to know what is in the network to make sure actually mitigate those risks.

Suzanne: That kind of also relates to the cascading effects. If I do not know what APIs I have, I cannot know why I am having effects from one of my APIs failing. Why don't you talk about that one, the cascading failures risk?

McKinley: Yes, that one is similar. One of the main things you can do with that is also making sure that your network is well documented to know what APIs are dependent on each other. Because, ideally, none of the APIs would be dependent on each other, but that is almost impossible to do. But making sure that you know which ones are dependent on each other, that can help make sure that if there is a loss of service in somewhere in the network, you can more easily figure out where that is happening. Then, also just designing the network to make sure that the APIs are as loosely coupled as possible.

Suzanne: Right. Then for the first risk?

McKinley: Yes, for third-party integrations, for one thing, you can do risk assessments on all third-party software that you might be integrating into the network, making sure that code does not have serious vulnerabilities. Then you can also stay up to date on new vulnerabilities that come to light with the integrations you have in your software. There may be vulnerabilities that you do not know now, and that nobody really knows of right now, but they might come to light in the future. Making sure that you stay on top of that is also a way of preventing third-party software integrations from causing serious issues.
Suzanne: This one also, in my mind, involves a training activity because

modern software engineers just innately go out on the Internet and find snippets of code that are similar to what they are trying to work on. It is a shortcut that unless you are in a secure area where you cannot get to the Internet, almost everybody takes. But that is third-party software, right? It may not be commercial third-party software or labeled as open source, but anytime you go out and grab snippets of code from the Internet, you are essentially introducing third-party software into your system. I have talked to several software engineers that just when I said that, they kind of went, *Oh.* Have you run into that problem as well?

McKinley: I have not run into personal issues with vulnerabilities with that type of code because I have not worked in cybersecurity very often, but I have worked with a lot of people who do take code off the Internet. I was a teaching assistant in grad school, and I taught computer science classes, and that is a very common thing for students to do. They are allowed to do it. At my university, they were, but we do not really teach enough about how that can be dangerous in terms of creating vulnerabilities in the software you are creating. Yes, you are right. Teaching people that just taking code offline or using third-party software integrations may produce risks and potentially huge risks for the software you are creating is something that we should teach more about.

Suzanne: Yes. In my day, we did not have that resource. You might have a friend that had something they had coded that you could go ask them to send you the source code so you could retype it. We could not hardly even FTP at that point. Yes, I started back in the day of punch cards. That is how old I am. Now, like I said, is it a whole different world in terms of what people have access to. I think that is one of the things in my whole talking to folks out in the world about cybersecurity. Is it not my main field, but because of these podcasts, I end up talking a lot more about it. There is just such a lack of awareness of some of these risks. I am really glad to see you writing about this and talking about it so that we can really make this something that people think twice before they just grab something and don’t know what its security implications are, which kind of leads me to the subject of transition. What are the resources available to me and the software engineers that are out there that want to do a better job of using APIs and making sure that the APIs they use don’t introduce unnecessary risk into their system, and where can I find those?

McKinley: Yes, so there are some articles written by the SEI. If you look online about APIs and more specifically about zero trust, which can also help in the protection of APIs. There is the SEI Zero Trust Collection, which is a grouping of webcasts, podcasts, blogs, all talking about zero trust. All of those can help in terms of implementing zero trust and just making sure network safety is prioritized.

Suzanne: For you, you are new, but you have already said you don't like to be bored. I am guessing you have something new and different that you are going to be researching. What are you working on that we can bring you back to discuss in a few months?

McKinley: Yes. I have been looking into more stuff about APIs, specifically about coding practices and general best practices when it comes to APIs as a whole.

Suzanne: Excellent.

McKinley: Hopefully, I will end up doing something like that in the future.

Suzanne: Also needed, something you can take back to your computer science classes you used to be an assistant for.

Suzanne: All right. I want to thank you, McKinley, for talking with us about this today. As I have said, I think this is one of the modern software risks that especially those of us that sort of come from my generation don't think about as often because we came at building software completely different way than is it being built today. I think is it important to get these ideas out to everybody, not just the young people, but all of the people that are working in this industry. For our audience, as I mentioned previously, we will include links in the transcripts to the resources we mentioned in this podcast. Finally, a reminder to our audience that our podcasts are available everywhere you find podcasts including my favorite, the SEI's YouTube channel. If you like what you see in here today, give us a thumbs up. Thanks again for joining us.