Building on Ghidra: Tools for Automating Reverse Engineering and Malware Analysis

Featuring Jeffrey Gennari and Garret Wassermann 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's Software Solutions Division. Today, I am very pleased to welcome to our podcast Garret Wassermann, a vulnerability analyst, and Jeffrey Gennari [Jeff], a senior malware reverse engineer, both with the SEI CERT Division. Today they are joining us to talk about Kaiju, not the monster-heavy genre of films and television, but equally monstrous series of tools that they have developed that allows for malware analysis and reverse engineering that helps analysts take better advantage of Ghidra, the National Security Agency's reverse-engineering tool. So, we have monsters killing monsters, and I love that.

Welcome to you both, and before we delve into that topic, could you tell us about your backgrounds and the work that you do here at the SEI? What is the best part of your job? Jeff, you have been a guest on our podcast series before. Why don't you start and tell our audience a little bit about yourself?

Jeffrey Gennari: Sure. I have been working in the CERT Program here at the SEI for many years. I have done just about everything that you can possibly do at CERT. I have been in vulnerability-analysis work, secure coding, malware analysis, operational work. Over that time one of the things I loved most about the institute is that you get to see not only the day-to-day problems in the government and work on them, but you also get to take a step back and think about the longer term trends, the deeper research that we get a unique opportunity to pursue here, given that we are supposed to be filling in the gaps. I just love that. It is very dynamic, and you always get to reinvent what you're interested in.

Suzanne: Actually, that is one of my favorite things about this place too, because I have done several different kinds of research areas, and that is unique about the SEI indeed. Garret, what about you? Why are you here? What brought you here? What keeps you here? What is it that's exciting for you here?

Garret Wassermann: I have been a member here at CERT for seven or eight years now, I think. I started as a vulnerability analyst understanding security bugs, security vulnerabilities in common software that people use every day or common hardware like home routers and things. I would work with both researchers and the vendors, software and hardware vendors, to better understand those vulnerabilities and how we can protect end users, and to make sure that patches were developed to fix those things and send out to the end users so that everyone could finally be secure as things are discovered. In all of that work in trying to better understand these vulnerabilities, you see over time trends and different ways that things can go wrong. So naturally you want to look more into the details and better understand those problems. That was what led me to eventually moving from vulnerability-coordination work we call it—coordinating patches and fixes—into executable-code analysis and malware analysis where I work with Jeff and others to better understand where these vulnerabilities come from and to identify them in software and to create tools that can help automate a lot of this process, because a lot of it is really difficult to do and very time consuming for analysts. So if we can try to find those patterns and things that link problems together, we can more quickly find and fix these issues and make sure that people are protected.

Suzanne: I appreciate that as a recent victim of a clickbait thing. The faster that we can get those sorts of things out of our systems, the better. I think that both of you exemplify this research-to-practice, practice-to-research kind of idea we have at the SEI. You have both been in practice, and now you're applying what you've learned to research. I know you'll be back out again in the practice field, because there's always more to learn here.

But today let's talk about Kaiju. This is a series of tools and capabilities that are built on Ghidra, the NSA's tool. I want to make sure that people understand why do we need another tool? Why isn't Ghidra enough? What is it that makes it important that we have Kaiju? What does it accomplish that is unique? Jeff, do you want to start us off with that one?

Jeff: Sure. Ghidra is an impressive toolset. It does what it is designed to do really well. It has got a compelling infrastructure where it has a fully functional decompiler. Just as importantly, it provides a really interesting programming interface, a programming API, so you can extend it. We've been working in program analysis, automated malware analysis, tooling for a number of years on various different platforms. The reason we wanted to incorporate Ghidra into that toolchain is because of the decompiler. It allows us to get access to higher level information to

make insights at a low binary level that you can easily apply to the source-code level, which opens the doors for all types of analysis.

Suzanne: So we are basically taking that core capability and then expanding it. It is sort of like a monster that has lots of different arms. We are creating the new arms that are going to do all the interesting things, and that allows you to deal with new threats. Because I think that is one of the things that we're always about in CERT, is we have to have flexible tools that allow us to bring the new threat in and analyze that as well. Is that one of the things that Kaiju helps us with?

Jeff: Absolutely. I mean, you can think of Ghidra in some respects as a platform for building new tools. So Kaiju really extends that paradigm where we have a good solid…Kaiju, I guess, means "many monsters," something like that. Garret can correct me on that if I am wrong. But there are lots of different kinds of Kaiju. If you look at what we have actually done, Kaiju itself is a platform for which we build additional tools that will tackle specific problems that we encounter in malware analysis.

Suzzanne: OK. All right, so, Garret, Kaiju is built on Ghidra. So you are not starting from scratch. But tell us, what is it like to build on someone else's toolset, somebody else's platform like that? And what kinds of collaboration did you get to engage in as part of doing that?

Garret: Well, one really great thing about Ghidra is that it's open source. So the fact that it's online and available to researchers allows us to actually talk with both developers of it, through GitHub and conversations and all as well as researchers that are using it in their practice. We can all bounce ideas off of each other and report bugs or suggest fixes, or whatever it is. So, open source really allows us to be able to foster that sort of collaboration. That is one of the important things that came up with Ghidra because before this point, so much of this analysis, as I had mentioned before, is done manually at a great expense in terms of time and energy and things. There wasn't a really big framework like this that had this complete level of debugger and things like that, decompilers Jeff mentioned.

So it is a very complete toolset, and one other reason I like it is, aside from the debugger and the decompiler, and all the more technical aspects of it, it also comes with a user interface, which is something that is very lacking for a lot of tools in this space. A lot of times they are very academic and technical and really hard to get working. So if we can develop tools that both are very powerful and help analysts that are also very easy to understand—you click the green button or something and it gives you your useful information. That's really a win-win for everyone. It allows us to perform our analysis faster, to find problems faster, to get fixes faster, all the stuff I was talking about. Building on top of it is just kind of understanding that framework, going through the documentation, asking questions of the developers and of the researchers, learning from what they've learned, and building out our own framework; kind of

understanding the chunks that are missing so far and adding to it to help build our tools. Then making that available for people to build on top of as well, so that we are part of a community collaboration.

Suzanne: One of the things that has characterized CERT from the very beginning is this collaborative aspect and this…What I see is, I call it *codification*. Kaiju, Ghidra represents codifying a bunch of knowledge that has been very, very hard earned in terms of getting it, as you said, Garret. A lot of these are very labor-intensive, knowledge-intensive, experience-intensive kinds of techniques, and being able to put them into a platform that allows people that are less experienced, less skilled, less knowledgeable to actually have good results and not make mistakes that especially beginners are likely to make, I see that as really bringing the state of the practice forward. It is sort of like when telephony went from having to have specialized operators to actually being allowed to have end users dial their own phones. We don't remember a time when you had to have an intermediary there. But in many ways, the malware analysis has been a very specialized community. I am thrilled to see some of this stuff becoming more available to a larger community because we all have needs in this area, and not everybody has a Garret or a Jeffrey on their staff to actually be able to tackle the really hard problems. I think, from a transition viewpoint, we are really making a big step forward with Kaiju.

So, this is not just on Ghidra, right? This is also part of a larger body of work inside of CERT called Pharos, and Jeff has talked with us about that previously. But, Jeff, would you just give us a little bit of an overview on that family of tools so people that are familiar with Pharos but may not be familiar with Kaiju yet can figure out where it fits?

Jeff: Absolutely. So, Pharos, I think you've hit a lot of the high points already. Reverse engineering, malware analysis has historically been a very manual process. You need to find people with a very specialized set of skills, get them in your employ, and then turn them loose. It's just not a whole lot of those resources out there. It doesn't operate in general at Internet speed. So, we set out years ago to try to find new and better ways to automate as much of malware analysis and reverse engineering as we could. That led us into a couple of different interesting directions. One of them was, people have been trying to gain insights about how software works without running it for a long time. So, in reverse engineering, we have humans do that. But in static analysis, program analysis people build tools that work on good formalisms to actually gain insights about a piece of software. So, what Pharos was, was an early attempt to try to merge those worlds where the static-analysis tools, program-analysis tools historically worked on source code to find defects, very specialized. Pharos was an attempt to apply some of the static-analysis techniques we observed in other domains to reverse engineering at the binary level.

A big change in the world since we started that was the maturation of decompilers. They just got better. Hex-Rays decompiler, Ghidra's decompiler, these are commercially viable tools now. So, we've been continuing to refine our program-analysis tools, but with better information. So, we get better insights, and so, Kaiju is the latest incarnation of that. Some of the tools in Kaiju are designed to be program-analysis tools that operate on the data that Ghidra produces as opposed to data we have to produce ourselves.

Suzanne: So, as Kaiju evolves and adds more arms into its monstrous toolbox—I am going to overuse that metaphor, but I love it—what are some of the areas that you are thinking about now that you are going, *You know, Kaiju is great, but what we really need now is X.* Garret, what are some of the things you're thinking about? And Jeff, I'd like both of you to answer that one. So, what are you thinking about next?

Jeff: Some of the problems that I have been very interested in are reachability-style problems. So, how do I get to a certain point in a program? With malware it is not uncommon to have any number of checks to prevent analysis. Detect it running in an analysis environment, detect the presence of a debugger, only run under certain circumstances. These are all constraints on what the software will do. We are doing this, but I would like to continue exploring new and better ways to use Ghidra and Kaiju to automatically determine a viable path to a piece of malware. If I want to get to the part of the malware that does the badness, whatever that is, what kind of inputs do I have to provide it? I want to give that information to an analyst so they can kind of separate the interesting bits from the crust that just goes into all software.

Suzanne: OK. Garret, what about you? What's on your mind in terms of things that you would like to explore further with Kaiju and Ghidra kind of as a platform foundation?

Garret: One aspect of the tools that started in Pharos that would aid in identifying and understanding malware is the use of what is known as fuzzy hashing, or the way that we used it is function hashing. It is a method of trying to identify unique aspects of software or bits of code that are shared between programs or maybe unique aspects of those programs for one thing to identify if a program is malicious to begin with. Then also to be able to compare them to say what aspects of each of these pieces of maybe some malware or ransomware, whatever it is, what do they share so that we can try to better understand, what are their capabilities? What are the things that they do that we should look out for so that we can better protect computers and understand what they are doing so we can remediate it? We ran into some limitations in doing that with the original Pharos implementation of that tool because of just all the changes in the way software works. Now there are many different programming languages that are used, and there's many different hardware architectures that are used. Windows on Intel 32-bit processors was like the dominant thing, but now everyone has arm processors on their phones, in their pocket and things like that now, right? We are seeing it expand to all these different

architectures, and that was one positive thing that we got out of moving to the Ghidra platform, that it already had a lot of support for all these different architectures involved in it. So by taking those methods and moving them over to the Kaiju platform using Ghidra, we are able to expand the reach of those tools. We are looking into how to better understand and evolve those tools into what we're seeing in modern software on different architectures and things like that. I am very interested in better understanding what some of our techniques that used to work for Intel processors, what that looks like on other architectures and other programming languages. We are seeing things go beyond C and C++, things being written in all kinds of different programming languages. At the bit level, the byte level, they look very different. They are compiled into very different code. But we can use the Kaiju and Ghidra platforms in order to, as Jeff said, decompile it and better understand the code. That gives us better hashing tools that can produce what is known as YARA files that allow us to give that directly to analysts and IT professionals and all to better understand their networks.

Suzanne: You are going to be busy for a while because I don't think that expansion of hardware architectures is going to slow down, and I don't think the expansion of languages is going to slow down. You are going to have your work cut out for you, which is good because we like to be busy, and we like to have cool problems to solve.

The other side of our work besides, *What's the new stuff that we want to deal with*? is transition. For people that are ready to play in the Kaiju sandbox, they are ready to start engaging the monsters, where do they go, and what resources are available to them? Obviously, Ghidra is open source. We have got Kaiju, Pharos. Where do they find these things and where do they find help in learning how to use them? Especially if they are not a malware expert themselves.

Garrett: Yes. Ghidra itself is available. It's on GitHub, but there's also a website, ghidra-sre.org that will point you to the GitHub website itself where you can download Ghidra itself. Then Kaiju is available through the CMU SEI GitHub page. You can download the source code and look at that directly or under the "Releases" page, you can actually just download a precompiled extension for Ghidra. You just point Ghidra at that and it installs it. Then you will get some pop-up window that installs it that says, *Hey, you have Kaiju installed now. Do you want to start using tools?* Then you can go from there and kind of play with some of the hashing technology and the pathfinding technology like Jeff was mentioning and see how that works for you. There are some built-in help manuals in Ghidra that we started writing to help guide new people. But, of course, having everything open source and out on GitHub means that it can be a lot more interactive and collaborative. Folks can always leave comments or bug reports or something and get the discussion going if they have any questions.

Jeff: And they do that. They do. We have built a nice little community around these tools.

Suzanne: Excellent. So transition. I am assuming you also will present about this to virtual conferences, it's not so much live anymore. But we have blog posts and other resources at the SEI in terms of guides that I also want to highlight. If you are my age and you listen to Garret's answer, it's kind of scary because he didn't say anything about tutorials or classes or anything. I'm from that generation that goes, *But I can't possibly start using it without some kind of instruction.* But we have some of those kinds of things too. In other generations it's much more important to get directly at the code, and I get that too.

I do want to thank both of you for talking with us today about this work and for being so creative in naming it and giving me some Friday afternoon entertainment. I am very happy that we are continuing in this path to making very, very complex, needed tools and skills and knowledge available to people that are not the world experts in this. Because I think that is the way that we actually manage security in today's world.

For our audience, I just do want to remind you, as always, we will include links in the transcript to the resources that we mentioned, the websites, etc. We will also have the blog posts and things that are available to you. So, I want to thank our audience for joining us today and talking about monsters killing monsters. I want to thank Jeff and Garret for sharing their insights into how these are evolving and are available for everybody to use now. So that's exciting. Thank you very much, and go forth and kill all those monsters.

Jeff: Thank you.

Garret: Yes, thank you.

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