Heather:

Welcome to the Hurricane Labs Podcast. I'm Heather. And this week we're doing a special double release to follow up on the Log4Shell vulnerability. So this is a two part series, be sure to check out Pentester's Perspective: Log4Shell in our links. Today, we're going to hear from a few of our SOC analysts about what they've been seeing from a blue team perspective. So thanks everyone for joining us. I appreciate you taking the time today. Why don't you go ahead and introduce yourselves and then we'll dive in.

Tony:

Hey everybody? My name's Tony Robinson. I am a senior analyst on the SOC team at Hurricane Labs. A lot of what I did with regards to Log4j was taking the detections that the rest of the SOC put together and our Splunk search devs, and it went from customer to customer to see whether or not there were any major issues as a result of Log4j. Additionally, I wrote up a blog post and did a small video demoing exploiting this on an older version of Minecraft with an older version of the Java JRE. I was legitimately surprised with how easy it was to exploit it once you had all the right things in all of the right places.

Kurt:

I'm Kurt Wolf. I'm one of the security architects for the SOC. I helped design and work on making the Log4J search that we used.

Josh:

Josh Neubecker. Also an architect, and worked on creating our detections around Log4Shell.

Dustin:

Dustin Miller. SOC analyst that helped put out the initial advisory and do some threat hunting.

Heather:

So the last time we talked, security teams had just learned about Log4Shell, less than a week earlier. Now, a little around a month has passed. So what have we learned about this vulnerability? Correct me if I'm wrong, but there haven't been any major breaches as a result of this, right? Does that mean the worst of the danger has passed or is the risk still there?

Tony:

To the extent, I've been following it on information security, social media, seeing what other researchers are talking about, seeing the things that they're discussing and to this point so far, I really haven't heard of any major compromises surrounding it, but it's one of those things that is going to be a forever day vulnerability. That's going to be there for a long time, lurking in the unkempt corners of both organizations shadows where a lot of their critical infrastructure lies or where some of their core software lies that perhaps might use Log4j and might not necessarily be publicly exposed to the internet, but might still be vulnerable regardless. So I have a feeling that it's going to be one of those things that'll be there for a while that they just aren't aware of because they might not have wanted to test it or see if their core applications were vulnerable initially, but I just have a feeling that, that's going to be one of those issues that we're going to be dealing with for a while.

Dustin:

I also feel like there's not going to be as much public disclosure of attacks that were started by Log4j simply because of the newsworthiness of the initial vulnerability. I think it was last week or over the weekend, there was a lot of exploitation seen for VMware Horizon that was going on and being reported about. And I just feel like moving forward, it's going to be something that companies won't want getting out. That it was cause, the initial exploitation was via Log4Shell, because it was such a big deal when it happened.

Kurt:

The tech on the end of all of it is, you also have to have some of the companies that probably didn't take the time to, like Tony said realize that they're actually vulnerable, that was taking place and may have just been like, "Okay, cool, carry it on." Or may not have even known what's going on. And those might be the companies that are compromised and don't even know it.

Tony:

Yeah, it's kind of funny because a little while back, I saw a meme, but like about people saying like how they were all worried about Log4j and testing their infrastructure for it. And then meanwhile, somebody suggested that there was a vulnerability and confluence from one of their Atlassian suites that they were hosting in house and that they got compromised with that because they were too busy with Log4j. So it's like one of those things where you're not really paying it. I guess the moral of the story is you're not really paying attention to it, but it's still there. Like as an example with me saying that you might not necessarily be aware of that you're vulnerable. It might be an internal only application or an intranet application that might not be publicly accessible, but when an attacker gets access to your network and they're trying to pivot, or if you have a penetration test going on and they're looking for ways to gain access to other systems, they're going to say, "Oh, this is a Java application it's running on Tomcat. Let's see if Log4j is vulnerable." And then they might just throw a Log4j payload at it and pivot to another part of your network that you might not want them to be on.

Heather:

What about some of the other Log4j vulnerabilities as the patches came out? Are there any things that people should like watch out for or any other issues that are cropping up as a sort of a ripple effect from this?

Josh:

One thing I did see wasn't necessarily vulnerabilities in later versions of Log4j, it was patches that were released by vendors who were using Log4j. I saw a couple vendors that published patches that first weekend, but the patches they published only had the workarounds, not the actual Log4j patches and those workarounds were later proven to be ineffective. So those vendor patches didn't actually resolve the issue while patching Log4j itself did.

Heather:

Got you. So what is it that makes Log4Shell so tricky? Why can't you necessarily trust that your log not showing a Log4j payload means that you are secure?

Tony:

Kind of a difficult thing to assess, it goes back to the conversation where you might not necessarily be aware that a system's vulnerable because it's in your intranet or it might not have been tested by your vulnerability scanners or vulnerability assessment team because it's considered a core part of your business or a place where attackers aren't expected to get access to from the public internet a place that they might have to pivot. So it's a hard question to answer. I mean even if it was patched, there are like, some people have noticed there had been a couple of issues with some of the patches, maybe not fully covering all of the use cases or maybe not covering cases where there are gadgets or other ways to get code execution other than telling the vulnerable Java application to go download this class file. That was the trust URL code based feature. So it's just somewhat difficult question to answer. I don't know if anybody else has anything to say about that.

Josh:

I think there's also a lot of obfuscation that can be done with Log4Shell that makes it difficult to detect. And also, where are you logging? Where do you even have visibility into where the Log4Shell payload is? It could be difficult, especially moving forward with that, because I think a lot of the low hanging fruit has been patched or resolved or there are detections around it, but the more interesting, difficult ways of exploiting this, like not just using the user agent field or something at an HTTP message that would get logged, maybe a post request or something gets logged or anything else it's just wherever you have input that can be controlled by an attacker that then gets logged by Log4Shell. That's where your attack service is.

Tony:

Yeah. I think one of our previous podcasts, I talked a little bit about that. Somebody was suggesting if you had a Prox smart card reader or if you had like a physical access system and you encountered a Prox smart card hub a Log4j payload on it that could be something. I was like, man, I didn't think that this attack service would be vulnerable, but if it's running Java, it's using Log4j, it serves as attack service. And it's just one of those weird corner cases that you don't think about.

Heather:

The word of the day for this vulnerability is ubiquitous, right? Because Log4j is in so many things like even some kitchen appliances can have Java. So now we've talked about security and IoT on our podcast before, but what's the big thing when it comes to IoT and security in the face of Log4Shell?

Tony:

Well, I know Java's big tag line is that billions of devices run Java. Well, there's two ways that you can take it. That there might be a lot more stuff in your home network that might be vulnerable to this than you might imagine. Or you could take it to me as like, well, it runs Java that doesn't necessarily mean it has Log4j on it, but that was a whole other thing as well. Like when people were poking Minecraft and managed to get code execution off of it, it was like, nobody thought that this Java base video game that mainly everybody plays it, but is mainly played by little kids. Nobody thought for a second that Log4j was a dependency on this game. So I personally really don't care for smart devices in my home, but then again, I'm somewhat of a Luddite when it comes to that stuff. I've got plenty of computers and I run my virtual machines on it and do all my reproduction testing and all of that stuff on it. But smart devices or like Alexa or smart cleaning devices, I think we have one robot vacuum in our house. And I was just like, don't you put that on the wifi. Just let it do its job. I'm not giving this thing an internet connection. So I guess it lines up with what your risk tolerance is. If you enjoy using smart devices and you think that

they add features to your life, that you really like, then you have somewhat of an elevated risk, but I really don't think very many of those devices are going to have Log4j on them or they're going to be using those vulnerable libraries. I guess, if there was anything to get out of that ramble is, just because it uses Java doesn't mean it's necessarily vulnerable. But it never hurts to check that your IoT vendors, security advisories, or their product updates page to see whether or not there's some sort of a firmware update or some kind of acknowledgement that it might be an issue.

Kurt:

And just to tech onto what Tony said, I mean, essentially you you're reliant on the third party patching it. I mean, if you're using their hardware and their software designed for it, you're relying on what they're doing, but just because it has Java, as Tony said, doesn't always mean it's running the vulnerability. So, I mean, I don't say end user, there's really not much you can do. It's either you use the products and just understand that any of the security patches are reliant on whoever you bought it from. And that applies to more than just a Log4Shell. I mean, that applies to any really hardware you're buying that is connected to the internet.

Heather:

So the FTC recently issued a warning to all companies that use Log4j and have yet to patch for it, including referencing the Equifax breach settlement of 700 million before making the following statement. "The FTC intends to use its full legal authority to pursue companies that fail to take reasonable steps to protect consumer data from exposure as a result of Log4j or similar known vulnerabilities in the future." So my question is, why might a company, or like Kurt was saying a third party not have patched this already given how severe vulnerability it is. If, you know, the risk isn't enough to make companies patch. Do you think this warning from the FTC is going to have much impact?

Kurt:

I think it's a matter of uptime and downtime of services. So if a company's able to have more uptime comparable to them getting charged or given a fee for not patching something, I think it all boils down to a business perspective. Does it make sense for me to have downtime on my servers for me to properly patch this? Or should I just take the millions of dollars thrown at me that I would've lost for being down for a day's worth of time anyways? I think it's all a balance of where they're going to save the most money, like anything in business.

Tony:

Yeah. I have to agree with that statement is a lot of companies look at it as a cost versus benefit calculus. I really hate to say this I feel like all companies should be responsible with the user data that they're entrusted with, but a lot of them just look it as a cost calculus. Is it cheaper for me to take the fees or take the being out of compliance with the FTC's ruling or recommendation, or is it more cost efficient for me to take the loss of potential customers or potential downtime and get the issue patched as soon as possible? Do I have high availability infrastructure so I can keep running while patching part of my infrastructure? It's that whole, it's just a cost benefit analysis for them. So they might be trying to fly under the radar and say, well, we didn't know that we had this or it's just a matter of what's the cheaper alternative to them.

Heather:

I was poking around on Reddit a while ago and I came across a debate that kind of touches into this. The question on this post was that should board members or the C level executives be held criminally liable for cybersecurity lapses and breaches? Should be... Especially given how very public Log4Shell was. I mean, is that a conversation worth having?

Tony:

Well, I've got some strong opinions on that because we all saw what was going on with the Equifax breach in what, in real time, how they had to testify before Congress and say, how did this issue happen? You were entrusted with so many people's information and credit data. And then they said, one of our IT admins was responsible. They were the ones that did it. And then it just shows that there is no responsibility up top and even if it does get pinned on an executive or anybody in the C-suite more often than not, that simply just means that they choose to part ways with that company and they get a severance package to move on to another company. Albeit much more quietly. I mean, that's just my general opinion. If anybody else has anything else to share or differing thoughts or anything else to add.

Kurt:

I thought we just blame the interns for everything. Isn't that what everyone does? Get more or less jokes, but that's my two cents on all of it.

Tony:

But even talking about that, I don't know how familiar everybody here is with the CISSP and all of that. I don't actually have the certification, but in the distant past when I was considering getting it and I was studying for it one of the key things that they always told you was that the upper management is usually responsible for whatever happens if it violates the law. So you got to make sure your ducks are in a row and that the people you're managing are actually falling in compliance and doing what they're telling you that they are doing in order to make them compliant. But ultimately if there's a data breach or something that usually falls up the chain. Who is responsible for actually making sure this is compliant? Who is their manager? Why weren't they following up? But as we see in the real world, that isn't always the case. We're talking about blaming the intern or blaming the IT guy for making sure it wasn't patched. Well, whose fault was it that they didn't follow up on that?

Dustin:

And I mean, often in these cases, there's always one scapegoat. Even if it is like the CISO or CIO or something. Usually it's the overall posture of the entire company, not just the one person who ends up taking the fall for it. Because I mean if you look at CISO how often in these times are they actually given the resources and support that they need to properly maintain the security of the company? And that often falls on the CIO and the CEO and other people, depending on who, how much money they end up getting. So blaming on one person is hard.

Kurt:

I don't think it's ever one person doing it, but my whole point of saying, oh, it's the intern or, oh, it's the one weird IT guy we don't like. It's easier to pin it on one person and quote, quote unquote, say we resolved and fixed the issue. We got rid of the individual who made the mistake or all of our policies are in correct, in order and they just happened to not follow it right. It's much easier to set a scapegoat like that and pin it and throw it out. But Dusty, I agree with you a hundred percent. There are so many moving parts that it's almost, I would probably say in most cases, almost every case, not pinnable on one

specific person. There's probably other moving parts that might have made that individual to make that mistake or something of the sort. And Tony, it's not that it's definitely the higher ups on, I mean, at a certain point you're as a, like the engineer or someone down more on the technical side of things, you're still not the one making the larger calls on stuff. You're the one acting out the orders. So the person giving the orders, like you said, from the CISSP and the courses you took and talking about that, I do agree it should fall on the general or the commander making the statements or the rules that people below should be following.

Heather:

I think what the FTC's warning does make clear going back to that statement is that this can have an impact on average internet users. So online shoppers, streamers, so on, people who can be impacted by a business's decision regarding Log4Shell and whether or not they patch, but avoiding exposure seems almost impossible. What steps can average users take to protect themselves in the face of this and other vulnerabilities outside of their control?

Kurt:

I think it's the basic end user precautions you need to take. The average person's not running a Log4Shells. They're probably running Java apps readily accessible from the internet, from their house. I mean, except for maybe Minecraft, which was where the stuff was found, but probably data ingestion and how you're setting your passwords and making sure the sites you're adding your data in is an over clear text and just basic knowledge for the end user of good security practices.

Tony:

Yeah. The blog post that you link from the Hurricane Labs blog, it's a pretty good practical information. There are, you as an end user you're outside of the scope of influence as to whether or not a company or a vendor you're using is patched properly if they patch it at all. But the things that you can do are maintaining situational awareness, like around your identity or around your credit card or your checking account information. I've got credit checking through my provider and all of that kind of thing. And it's just a matter of checking your balances every so often and question where certain charges came from. It's a case of being aware of your finances and where things are going. And then if you notice something strange, immediately shutting it down. That's what you have control over. And it's one way to keep yourself protected is to be aware of what expenses are coming in and out of your account every month.

Kurt:

Well it's sad Tony, it's that inconvenience of going through and changing that, or the inconvenience of changing your passwords on X amount of sites. I think that inconvenience itself is what holds a lot of people back from actually having good end user security practices, which is...

Tony:

Yeah. I mean, that's like a lot of the times why we find password reuse. It's because the culture doesn't tell, security culture at a company just tells you, you have to go to security training and they say, you can't reuse the same password. You got to change it every 30 to 90 days or whatever the NIST standard seems to be. But nobody at the company ever has a sit down and say if you do this, if you use the same password, this also has repercussions in your personal life. If you use the same password for say, logging into your American express card site, or that you use to log in from Netflix some could possibly get password data, or there might be a compromise of Netflix or another streaming service. And then

they're going to reuse those credentials and try and see if they can get access to anything else. Nobody ever puts it into that terms and say, this is why you should do as a password manager. And this is why you should try and take advantage of multifactor authentication whenever it's available. Sure. It's a pain in the butt to have your phone and have to type in those numbers. But the alternative is you have to reset your passwords, or you have to worry about your bank account or your credit card being maxed out.

Josh:

Yep. And you can prioritize there. You don't need to go changing every single one of your passwords immediately, but just focus on your financials on your email. Make sure you have two factor on these and make sure they have secure long passwords that are all different.

Kurt:

That's The biggest thing. If you're using the same email, which I assume a lot of people have one main email, probably a Google account, or if you're older, you have a Hotmail account or something. If you're doing that, you're logging into 10 different sites. And if you have the same password, that's an easy access to probably your social media, your bank accounts and et cetera.

Heather:

My husband still uses his Hotmail. I had someone give me an AOL email address the other day.

Tony:

Oh goodness.

Heather:

Yeah. Oh, wow. I think this one's opening up a whole nother can of worms that we may not go down rabbit hole but talking about two-factor authentication and the security of text messaging for that purpose.

Kurt:

What like using SMS for two-factor?

Heather:

Right. Yeah.

Tony:

Yes. So I've got opinions about that. And a lot of people say, yeah, there are some security people out there that say if you're using SMS for two factor auth, then it's worse than having nothing at all. And I sincerely don't believe that. If it's the only option available to you, NIST really doesn't recommend it or other cybersecurity standards don't recommend it. But if it's the only offer or option that your bank or your credit card company's giving you, if they don't want to give you an actual, two factor auth program, or be able to put something into your Google authenticator, then I'd rather take it and have it than not have it at all.

Heather:

Something's better than nothing.

Tony:

Exactly.

Heather:

So our pentesters talked about a few tools that they use to identify possible Log4Shell vulnerabilities, including Burp suites, Log4Shell everywhere, and NSE script and ZAP. Are there any other tools or strategies that you can recommend to help address this vulnerability?

Tony:

I really don't have anything easy to add to that, but if it's in any kind of an application where the user is going to have the ability to input data, it's one of those things that's a much easier said than done thing. But if it's any kind of a web app or any kind of an internet app or anything that a user is going to have the ability to control the input, you're going to want to probably send a Log4j payload to it, to see if anything interesting gets logged elsewhere. See if like there's any point where it goes back to the point that we made earlier. Do you know where these logs are going? Do you know where they're being collected? But also do you know where are the points where the user can actually input data? And is that data validated in any way? That's about all I really got. I mean, it's one of those things that's painful to enumerate every point where while this is the user has the ability to influence what's going on here because they can put this in this input box or that, in that input box, but is it validated? Is it checked? And I mean, I know I'm starting to sound like I'm rambling here, but that's the gist of it. Aside from testing out your applications and using ZAP or using C scripts to do touches here and there to see if just sending a payload or sending a Log4j payload and user string, or as a part of like an HTTP request. The other alternative is to start testing your inputs.

Heather:

All right. Thank you so much for taking the time to talk about this and for sharing those tips, this that's all the time that we have for now. So listeners be sure to check out our links for the resources that we've mentioned today, and until next time, stay safe.