Spring4Shell

Description: We'll wrap up this week's podcast by revisiting Spring4Shell. Last week, when we first mentioned it, it was just a questionable itch. Now, a week later, it's a full-blown outbreak deserving of today's podcast title. But before we roll up our sleeves for that, we're going to examine credible reports of a zero-day in the Internet's most popular web server platform. We're going to take a look at Microsoft's newly announced "Autopatch" system, and the rapidly approaching end-of-security life of some Windows 10 editions. We have another instance of an NPM protest-ware modification of a highly used library, and I want to share a bit of miscellany and listener feedback. Then we'll finish by looking at what one week has done to Spring4Shell.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-866.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-866-lq.mp3

SHOW TEASE: It's time for Security Now! Patch Tuesday edition. Steve Gibson is here. We're going to talk about Spring4Shell. We talked about it as a concept last week. Well, the concept one week later is now a reality. We'll also take a look at a problem, a zero-day perhaps, in Nginx, the most popular web server on the web, on the Internet; and Microsoft's newly announced Autopatch system. It's all coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 866, recorded Tuesday, April 12th, 2022: Spring4Shell.

It's time for Security Now!. I know you've been waiting all week long. He's finally here, Steve Gibson of GRC.com. He actually is in this TV all the time.

Steve Gibson: It's lonely in here.

Leo: All by himself.

Steve: I've got some blinky lights to keep me kind of entertained. I wait for something to happen. Every week it turns on, and I go, oh, hello.

Leo: Hey, I'm on. No, not true. Steve is a very busy guy. In fact, we're very grateful that he takes the time to do this show every week. I hope I say that enough. Thank you.

Steve: It's been a very good thing for my life, Leo. I do not regret it for a moment.

Leo: Me, too. Me, too. I feel the same.

Steve: So we're at Episode 866. I think I've got the number correct on the show notes this week. I didn't advance it last week, and then I always am reediting the same doc file. So after I stripped everything out of this single doc file last night, I thought, ooh, I could have changed the number and made another PDF, and the show notes would be correct forever. Now they will be wrong forever. But I guess I could get some Scotch tape or something.

Anyway, this is April 12th. It is Patch Tuesday, which is apropos of the Picture of the Week, which we will show in a minute. We're actually going to wrap up this week's podcast by revisiting the first topic of last week's podcast. That was when we mentioned what was at the time a somewhat questionable itch. Now, a week later, it's a full-blown outbreak, deserving of the podcast's title. That is to say, Spring4Shell is no longer just theoretical.

Leo: Ugh.

Steve: But before - I know. It's just - it's amazing. Before we roll up our sleeves for that, we're going to examine credible reports of a new zero-day in the Internet's most popular web server platform. And that's not where you want to have a zero-day. We're going to take a look at Microsoft's newly announced Autopatch system and, like, what is that all about? And the rapidly approaching end of security life for some Windows 10 editions. We have another instance of an NPM protest-ware modification in a highly used library. And I'm going to share a bit of miscellany and listener feedback before we plow into taking a look at what one week has wrought in this next Spring4Shell vulnerability. So a lot of interesting stuff to share with our listeners.

Leo: It's going to be another thrilling, gripping edition, as the announcer says, of Security Now!. Very excited, as always. Thank you, Steve. On we go. Your turn, Steve.

Steve: So our Picture of the Week, it looks like somebody actually made this shirt. It looks like a photograph of an actual T-shirt. It's a black T-shirt. It's got the well-recognizable Windows logo, you know, the four different-colored squares. And all it says on it, it's got the Windows logo, and it says "Exploit Wednesday."

Leo: It's the day after Patch Tuesday.

Steve: It's the day that follows Patch Tuesday. So it's the answer to the question, what follows Patch Tuesday? Well, Exploit Wednesday.

Leo: Nice. Nice.

Steve: And, boy, you know, we just cover story after story, concrete example after example of how that's exactly true. And we're going to see a couple of those today.

Leo: This first one scares me because this is what I use.

Steve: It's what I use, yes.

Leo: Yeah, everybody, practically.

Steve: So for those who don't know, Nginx is spelled N-G-I-N-X. It's a web server and more that's been steadily growing in popularity. When I installed GRC's GitLab instance on a FreeBSD Unix box, Nginx was, and is, providing that platform's web services. Apache, which like forever, for like decades, was the leader, is no longer so. Nginx is now the de facto web platform for new installations, and it's now the most commonly used web server on the Internet. It's got a 33.2% share overall. And of the top 1,000 sites, it's just shy of half. It's 45.2% share of the top 1,000 sites. And it can serve as a reverse proxy, a load balancer, a mail proxy, and an HTTP cache. So lots of different things it can do.

And over time, large projects tend to get pushed to do things they were not originally designed to do. New chunks get added onto them here and there. And what might have once started out as an elegant and straightforward architecture ends up becoming awkward, riddled with special-case exceptions, and it becomes increasingly difficult to maintain. And of course, as we know where security is important, simplicity is really key. So in other words, these big projects almost inevitably begin to show their age.

And one such example is OpenSSL. As we've covered here, it's become so old, cumbersome, and creaky that various lean and streamlined alternatives have been built. It remains an amazing toolkit. I use it at the command line from time to time to do stuff with certificates that you just - there's like no other way to do them conveniently. But if all one wanted now was a fast, clean, and simple way of getting secure TLS connections, OpenSSL may no longer be the best choice.

So what of web servers? I thought it was interesting that the official Apache.org site claims, they said: "Co-founder Brian Behlendorf first came up with the name 'Apache' for the server. The name Apache was chosen out of reverence and appreciation for the people and tribes who refer to themselves as Apache." Okay, well, I think that indigenous Native Americans are a great and noble people. But those of us who have been around for a while will recall that that's not the case. The Apache web server got its name because, literally, it was "a patchy web server."

And the Internet archive doesn't lie. I've tracked it down. Question #4 on the Apache's FAQ from July 9th of 1997 asks: "Why the name 'Apache'?" And it provides the answer. It said: "A cute name which stuck. Apache is 'A PAtCHy server.' It was based on some existing code and a series of patch files." Okay. So my point is that it did get rewritten at one point when the world became aware of how important it was going to be to have a strong, robust server. But it's getting a little bit long in the tooth, and we have a new kid in town.

F5 Networks, which focuses upon web application security, needed a platform. So just over three years ago, in March of 2019, they purchased Nginx for $670 million. Which, again, we've talked before about how interesting it is that you buy an open source thing for that much money. It's like, uh, okay. Anyway, as a result, today it is they who are

investigating a credible-appearing report of a zero-day in Nginx. And again, there are credible-appearing reports, which we'll talk about in a second, including reports of successful breaches, using a zero-day in what is now the Internet's most popular web server and in use by nearly half of the top 1,000 Internet sites.

A spokesperson asked by the press yesterday, on Monday, said: "We are aware of reports of an issue with Nginx web server. We have prioritized investigating the matter and will provide more information as quickly as we can." Well, okay, that's good.

Now, the problem first surfaced on Saturday when a Twitter account connected to a U.S.-based group known as BlueHornet tweeted about an experimental exploit for Nginx v1.18, which is the current release. The group tweeted, they said: "As we've been testing it, a handful of companies and corporations have fallen under it." They didn't respond to requests for further comment. But a different researcher shared a conversation they had with the people behind BlueHornet about this issue.

The group explained that the exploit has two stages and starts with an LDAP injection. LDAP stands for Lightweight Directory Access Protocol, and an LDAP injection is an attack used to exploit web-based applications that construct LDAP statements from whatever it is the user supplies. And so if there's some tricky way of supplying some information that can in some way abuse what LDAP is doing with that user-supplied information, that's your way in.

BlueHornet said that they would share the issue with the Nginx security team through HackerOne, presumably for a bounty, which in this case seems fair if they've got a zero-day for Nginx, or through F5's internal platform. And BlueHornet later created a GitHub page where they explained in detail how they discovered the issue and how it works. For anyone who's interested I've got a...

Leo: Sounds like it's an LDAP exploit, though; right? I mean...

Steve: Yes.

Leo: Not necessarily a flaw in Nginx. You have to exploit LDAP first.

Steve: I think that's the case.

Leo: Yeah.

Steve: So they wrote: "We had been given this exploit by our sister group, BrazenEagle, who had been developing it..."

Leo: Okay.

Steve: I know.

Leo: That's not their real name, for sure. Okay, go ahead. That's as bad as Apache. Okay, go ahead.

Steve: Yes. Nor is it an Indian name.

Leo: No.

Steve: Or, they said, at least since Spring4Shell came out. Although this bears no relation to that. Spring4Shell we'll get to later. That's a Java problem. They said: "We are still in the early stages of usage and understanding it, as we are working on another vendor vulnerability." So, you know, what are you going to do? You've just got too many vulnerabilities to handle at once.

They said Gitworm - and that's that other entity that shared some details. "Gitworm was allowed to share that information with permission from our DMs. We were initially confused as LDAP doesn't [does not] interact much with Nginx. However, there is an LDAP-auth daemon used alongside Nginx which allows for this to be used. It primarily," they wrote, "is used to gain access to private GitHub, Bitbucket, Jenkins, and GitLab instances." They said: "Further testing is required in due time." So that was how they opened their GitHub posting.

Then they posted Update #1: "As some further analysis is ongoing, the module related to the LDAP-auth daemon with Nginx is affected greatly." Then we have a smiley face. They said: "Anything that involves LDAP optional logins works, as well. This includes Atlassian accounts. Just working out if we can bypass some common" - and they said WAFs, so that's Web Application Firewalls.

And they said: "Default Nginx configs seem to be the vulnerable type, or common configs. We highly recommend disabling the ldapDaemon.enabled property. If you plan on setting it up, be sure to change the ldapDaemon.ldapConfig properties flag with the correct information, and don't leave it on default. This can be changed until Nginx respond to their emails and DMs." And which is a bit of a story. They were having a bad problem getting a hold, getting anyone to respond from F5 which, you know, I don't understand that.

"Update 2: Been talking to some infosec people about this, some mixed responses. Some are saying it's a problem with LDAP itself and not Nginx, while ldapDaemon isn't always used. The exact quote is 'CI/CD pipeline hardens the instance. One of the steps is to completely strip out the LDAP module.'" And they said: "This is partially correct. In fact, it's an option when compiling Nginx. However, it could be a problem with LDAP itself. The issue with this is that it only works with Nginx instance using LDAP, such as any login portal that supplies that authentication method. Further analysis and testing is required." They said, again: "Looks to only be affecting this version. If it affects updated versions of the LDAP protocol, then we'll see what comes of that.

"Update 3: I (as I'm still in ATW, but I'm just the only one online) have forwarded our own questions, community concerns, further testing questions to BrazenEagle via email. They have yet to respond, as they are U.S.-based. Hopefully they can provide some further answers." They said: "While I'm still skeptical on the workings of this issue, it would explain the companies that were breached in under an hour during testing by BrazenEagle. They stated that they had passed this exploit to us as they were 'working on more lucrative exploits.' What that means, I'm not sure," this person posts. "Some few individuals were clearly told about this being in the works some months ago via Telegram chats, which maybe perhaps some Twitter infosec people were talking about it and ATW."

Update 4 briefly says: "Regarding what people have been suggesting on Twitter and on the issues page about this only being a LDAP issue, the problem with this is, during testing phases, it's only working on Nginx, not on Apache or other web servers. Also, Nginx have still not" - oh, yeah, also Nginx have still not replied. They really meant F5. Anyway: "DMs or email. We've emailed some companies that are affected that we've not breached, since that's," they said, "heavily against our ideals for support on the matter regarding security around this exploit."

The fifth update: "So we've been followed by an employee of Nginx on Twitter. Threw them a DM asking about the situation. No response yet. We've been working on another exploit for MongoDB and another database management framework. Looking to have the proof of concept out in a week's time. Video as well. Will be working on it with [this Gitworm guy] on Twitter."

And then finally: "Update 6: We got a DM from Nginx on Twitter regarding the issue." And I grabbed a screenshot of this Twitter dialog. They originally said: "Hello. Does Nginx have a vulnerability disclosure program, or a bug bounty program?" The replay back was: "Please report any security-related issues concerning Nginx to," and then they have "mailto:security-alert@nginx.org." And then the hackers replied to that DM: "Already have done. Did you get the email? Is there a template you wish us to follow?"

So, and then lastly Nginx tweeted: "Addressing Security Weaknesses in the Nginx LDAP Reference Implementation." And their tweet included a bit.ly link. And so the end of this GitHub stream is the eighth update, reading: "As Nginx have now released a blog post about the public releases of information, we've emailed them with a description, some familiarities of the issue that they highlighted over and assets affected. However, people are quick to jump on the 'This is fake' or 'This isn't anything' bandwagon. As we got no answer to if there is any bounty offered by Nginx for the findings, we've not shared any deeper information about this. If there's no bounty or even reward, we've looked at other options that would be to sell the exploit on either breached.co, exploit.in, or other sites."

Leo: See, that does make me suspicious. That's kind of blackmail almost; right? Now they're saying...

Steve: Yeah, well, and I notice that that apparently is not against their ethics, although they're saying that using it directly is. And then they said: "We've been offered about 200K in XMR" - which is Monero - "for the exploit." And then they finish: "If you're thinking that we're only interested in money," then they said, "yes, what do you expect? We're a threat group, LOL." So okay. Take that for what it's worth.

As for their part, Nginx is playing this as though they don't think that this is much of a threat. Their posting about this is titled, as I mentioned, "Addressing Security Weaknesses in the Nginx LDAP Reference Implementation," and it starts out saying: "On 9 April 2022, security vulnerabilities in the Nginx LDAP reference implementation were publicly shared. We have determined that only the reference implementation is affected. Nginx Open Source and Nginx Plus are not themselves affected" - just as you suggested, Leo - "and no corrective action is necessary if you do not use the reference implementation."

Leo: Do they mean the config file? What do they mean by "reference"?

Steve: Well, I got a kick out of this because in other words, "You weren't dumb enough to actually use the sample code of the way this should be used, were you?"

Leo: Yeah, that would be a bit - who would do that? I mean, c'mon.

Steve: And, you know, we've seen this over and over; right? The classic example from early in this podcast was when Intel published a reference implementation for UPnP which all router vendors naturally copied and pasted into their code. Then, when it was later found to be horribly defective, Intel said: "Well, we didn't mean for you to actually use it. We just offered it as a reference." Right. Okay.

Leo: Okay.

Steve: So Nginx said: "The Nginx LDAP reference implementation uses LDAP to authenticate users of applications being proxied by Nginx." Okay, right. "It is published as a Python daemon and related Nginx configuration at" - and then they have a link - "and its purpose and configuration are described in detail on our blog," and another link. And the blog is "nginx-plus-authenticate-users."

They said: "Deployments of the LDAP reference implementation are affected by the vulnerabilities if any of the following three conditions apply. Below, we further discuss the conditions and how to mitigate them." So the three conditions are command-line parameters are used to configure the Python daemon. Which is what they do in the reference implementation. Or there are unused, optional configuration parameters, as there are in the reference implementation. Or LDAP authentication depends on specific group membership.

So, and they finish: "Note: The LDAP reference implementation is published as a reference implementation and describes the mechanics of how the integration works and all of the components required to verify the integration. It is not a production-grade LDAP solution. For example, there is no encryption of the username and password used for the sample login page, and security notices call this out."

So, okay. At this point we'll have to see how this all plays out. Nginx has a corporate interest in, to some degree, I mean, they have to take responsibility, but they would like to downplay it. And they appear to be sliding the responsibility for this mess onto the shoulders of those who actually implemented their reference implementation.

But for all of us here, there's a larger takeaway lesson, I think, to be learned. It's sort of a variation on the tyranny of the default; right? And that is, never, for the sake of simplicity or clarity, offer an insecure example or reference implementation that you're not prepared to have pretty much everyone use, for real, exactly as it is offered without modification in a real production environment. Right? Because that's exactly what's going to happen.

Everyone's in a hurry. No one's as much in love with your stuff as you are. No one else knows it as well as you do. They don't want to make a career out of setting it up. They just want to get it going, you know, install it. Okay, fine, it works good. And then move on to the next thing. So it's necessary to assume that all of the default settings are going to be left as-is, including any code or examples that are provided as samples of this is how you set it up. Because that's what's going to happen.

Leo: Yeah, I mean, that's, when I set it up, that's exactly what I did. I used their conf, their example conf, I'm sure.

Steve: Yes. And I followed a recipe when I set up GitLab. And Leo, this thing, I just shudder when I think of what's going on. It's got so many moving pieces. I'm typing, it's like, you know, you're the Sorcerer's Apprentice, and you're typing, you're casting spells.

Leo: No idea what you're doing.

Steve: Into the console.

Leo: Yeah, yeah.

Steve: You know? And you don't even dare type them yourself, so you copy it out of the recipe and paste it over here, and you hit ENTER. And then the screen scrolls.

Leo: I'm glad to know that you feel that way, too, because I just assume that I'm an idiot and that I just don't know what I'm doing.

Steve: Oh, nobody knows. No.

Leo: Yeah. No.

Steve: And like when I compile stuff, you just see like the compiler is just gone...

Leo: Look at the makefiles. Who even knows what all those settings, all those parameters mean? I have no idea.

Steve: No, and this is why Windows doesn't really work anymore. They press "Build," and they just stand back. It's like, okay, you know. And then, like, does Notepad run? Okay, thank god. So on Sunday the hacking group claimed that they had tested the zero-day on the Royal Bank of Canada...

Leo: Ooh, oopsies.

Steve: ...but didn't explain whether the bank had actually been breached. It later said it did breach the systems of the Chinese branch of UBS Securities. Neither of those institutions responded to requests for comment from the press. But none of us should be surprised if we learn in the coming weeks, or maybe it'll be the title for next week's podcast, right, of sites breached by leveraging this newly uncovered zero-day in Nginx's reference implementation of LDAP-based authentication. Clearly, some people have just used it the way Nginx said, "Here it is." You know? Don't use it. But, well, if it's here, why should we not use it? So that's how the world got UPnP from the beginning. Remember back then in the beginning of the podcast, Leo, when it turned out that Intel posted something that you should not use?

Leo: Oh, yeah. Don't use this.

Steve: It was in every router.

Leo: This is how you don't want to do it, a reference implementation. How you don't want to do it.

Steve: Wow. Let's take a break. I'm going to wet my throat, and then we're going to talk about Microsoft's new Autopatch system and have some more fun with that.

Leo: You know, no one can understand everything that they're doing. And, you know, there are some risky things that everybody does. Whenever I, you know, what you don't want to probably do is copy and paste a cURL command from a website to install their software. But, you know, a lot of software you feel like, well, I know this site. I'm sure it's okay. It's executing a shell script with often admin privileges on your machine. We do that. I look at makefiles, they're hundreds of lines long. You're supposed to, when you're updating in Linux, when you're updating from the user repositories, you're supposed to read the scripts ahead of time to make sure they're not doing anything malicious. Ain't nobody got time for that. C'mon.

Steve: No, it's really true. And when you think about it, I mean, this podcast is getting to me, I have to say, because I'm, like, I'll be looking for some utility somewhere, something that does something, and I find it. And it looks good. But, you know...

Leo: But do you dare use it?

Steve: Exactly. You know? I want it. I don't have time to write it by myself. So, well, you know. And so what I do now...

Leo: That's why I like open source, because then somebody else can be looking at it.

Steve: I find that I am dragging and dropping more things on VirusTotal these days.

Leo: Yeah, yeah.

Steve: I'll get something, I'll just go, what do you guys think?

Leo: That's probably a good idea. At least do that, yeah, yeah.

Steve: And if it lights up like a Christmas tree, oops, maybe not.

Leo: Yeah, yeah. We do the best we can. But honestly, it's such a complex world.

Steve: It's really escaped everyone's control.

Leo: Yeah, yeah, yeah.

Steve: Okay.

Leo: Okay. All right.

Steve: A bunch of the tech press covered the news of Microsoft's new Autopatch system. And some noted that it was going to make Patch Tuesdays much less exciting. Because, you know, it's excitement that you're hoping for whenever Microsoft updates your system. Like, "Where did all my desktop icons go?" Okay. No one ever said that using Windows was boring. That's not something you hear often.

Lior Bela, a Senior Product Marketing Manager at Microsoft, explained. He said: "This service will keep Windows and Office software on enrolled endpoints up to date automatically, at no additional cost." This is still him. "The second Tuesday of every month will be 'just another Tuesday.'" Right, like the Pandora's Box that it's been recently. Or other than, unlike the Pandora's Box that it's been recently.

So, okay. What is this? It's going to be interesting to see how this goes. Microsoft explained - because, you know, I thought we already had automatic updates; right? Autopatch. Okay. Microsoft explained that: "Windows Autopatch manages all aspects of deployment groups for Windows 10 and 11 quality and feature updates, drivers, firmware, and Microsoft 365 Apps for enterprise updates. It moves the update orchestration from organizations to Microsoft, with the burden of planning the Update process, including rollout and sequencing, no longer on the organization's IT teams."

Okay. But anyone who's been, like, around in the industry knows that the whole point of giving control to IT teams was to allow them to carefully roll out Windows updates to enterprise machines only after first vetting those changes to make sure they didn't break anything mission critical. And yes, it's a big pain in the butt. But it's proven to be necessary over time since Windows updates have established such a track record of breaking things. Like you roll up an update, and now nobody in the organization can print anymore. That might be a problem.

So, okay. What's Autopatch? How does it work? What does it do? Microsoft plans to automate the process that IT teams have been performing for themselves in-house. The service automatically divides an organization's entire population of Windows machines into four groups known as "testing rings." Microsoft likes their rings, so we have more rings now. We have the test ring, the first ring, the fast ring, and the broad ring.

Leo: No. Really?

Steve: I'm not kidding you.

Leo: Oh, man. I want to be in the broad ring, whatever that is.

Steve: I want the broad - yeah. If it's last, that's where I want to be.

Leo: Yeah.

Steve: So you got test, first, fast, and broad. The "test ring" will contain a minimum number of devices. The "first ring" will contain about 1% of all endpoints that need to be kept up to date. The "fast ring" will have around 9%, and the "broad ring" will have the remaining 90% of all devices. So a few in the test ring, 1%, 9%, and 90%.

Lior Bela said: "The population of these rings is managed automatically. So as devices come and go, the rings maintain their representative samples." Samples. "Since every organization is unique, though," he said, "the ability to move specific devices from one ring to another is retained by enterprise IT admins," even though he started off by saying the population of these rings is managed automatically. So I guess that means automatically unless someone moves something somewhere because they don't want it in that ring.

Okay. Once these testing rings are set up, updates will be deployed progressively, beginning with the test ring and moving - presumably, if it doesn't melt down - and moving to larger sets of devices following a validation period through which device performance is monitored and compared to pre-update metrics. Which there's some corporate speak for you. So it's like, huh. All I'm getting now is a blue screen. I don't think this compares favorably to my pre-update metrics. What do you think?

So anyway, Microsoft announced that this new Windows Autopatch service will be released this summer in July. Either way, the good news is it won't bother non-enterprise end-users, like hopefully most of us, since it will be a new managed service offered for free to all Microsoft customers who already have a Windows 10 and 11 Enterprise E3 or above license. Whatever that is. If you have one, you probably know.

Okay. Now, the good news is, Autopatch includes "Halt" and - I wanted to say "Halt and Catch Fire," but no - "Halt" and "Rollback" features that will automatically block updates from being applied to higher test rings or rolled back automatically. Okay, that's good. But listen to this one. The product manager said something that I had to decode. He said: "Whenever issues arise with any Autopatch update, the remediation gets incorporated and applied to future deployments, affording a level of proactive service that no IT admin team could easily replicate. As Autopatch serves more updates, it only gets better."

Okay. What I think he said was that, when Autopatch breaks something, it learns about that breakage and doesn't do it again. What, on that one machine? Or on any of that enterprise's other machines? Or on similar machines globally? This is beginning to feel like more of that "We don't know for sure where Windows 11 will run" hocus pocus. Like where did all of the actual computer science go?

It sounds like Microsoft is using their telemetry feedback, and the fact that updating their operating system has become so problematic that they're going to turn all of the machines owned by all of their Enterprise customers into a gigantic neural network of "let's try this and see what happens." Anyway, I've never felt so happy to be a lowly end-user. This is going to be interesting. And I'll be listening to Windows Weekly to see what Paul and Mary Jo think about this because, wow.

You know, I guess if you had sort of a mid-size enterprise that didn't have the excess revenue to staff this kind of IT admin team that you now need, as evidenced by this, I mean, this is responding to need; right? So where every second Tuesday of the month this team stops their regular business in order to figure out what this month's updates will do to their enterprise, and so they've got a set of representative machines, you know, endpoints, running their corporate stuff.

And so they first install the updates there, and then see if everything works. It's like, did this break anything? Can we still print? Can we log in? Does our app go? And if so, then I'm sure they hold their breath, and they say, okay, let's roll this out to the fourth floor and see if it survives. And if so, then they continue. And so it's interesting to me that Microsoft has decided, okay, yeah, we're going to do that now. Just Autopatch. It ought to be Autoprayer. Anyway, we'll see. It'll be interesting to see what happens.

As I mentioned, we have another instance of Russian protest that has appeared in JavaScript's open source repository. On March 17th, so today's the 12th, almost a month ago, the Russia-based developer Viktor Mukhachev, who's also known as "Yaffle," and since that's much easier we'll call him Yaffle, altered his popular NPM library known as "event-source-polyfill." This change, which was introduced into v1.0.26 of "event-source-polyfill," will cause web applications built with this now-latest version of the library, and it's still current, by the way, nearly a month later this is still in place, it will cause web applications built with this update to display antiwar messages protesting the "unreasonable invasion" of Ukraine, to Russia-based users 15 seconds after a webpage which incorporates this code is displayed.

Okay, now, "polyfill packages," we might also call them "backfill packages," but polyfill's their official name, implement sets of newer JavaScript features on web browsers that do not yet support them. In this case, the "event-source-polyfill" package that's been deliberately polluted by its developer implements the very useful JavaScript "EventSource" API. This API allows a webpage to open a persistent connection back to an HTTP server, which then sends events to the browser. And it's a one-way connection which remains open until it's explicitly closed by calling EventSource.close() function.

So what's interesting is why anyone would need to backfill this particular API, since it's been present in all major browsers for quite a while. It was first adopted by Chrome and Firefox - get this - in their respective version sixes.

Leo: Whoa.

Steve: I didn't even - yeah.

Leo: We're up to 100 now.

Steve: Yes. Firefox is at 98, and Chrome is at 100. But what's interesting, Leo, is the chart showing the API's adoption profile had a single interesting and glaring exception: Internet Explorer. And it is probably the case that Russia remains the surviving bastion of Internet Explorer use.

Leo: Oh, isn't that funny. Oh, my god.

Steve: And by the way, Leo, if you didn't see Colbert last night, oh, goodness. He had something - they resubtitled Putin talking to the camera.

Leo: Oh, that'll be fun. I will watch that.

Steve: It was quite good. He was trying to raise money for Russia and was, for example, offering a box puzzle of, and he said, "with only four pieces missing." And oh. And that's just a tip of the iceberg. It's, you know, Colbert and his writers at their finest. So anyway, in order for Russia's inventory of IE to be able to run web applications that rely upon the EventSource API, that support needs to be "polyfilled," provided by this library.

But given how pervasive the use of Yaffle's "event-source-polyfill" package is, and given that it's only needed by IE, since all other browsers have incorporated the API natively for years, I mean, I didn't go back to figure out when version 6 of Firefox and Chrome were. But, I mean, it was a while ago. It must mostly be due to other developers not having yet proactively removed it from their own package dependencies because, get this. It is currently used by more than 135,000 GitHub repositories. 135,000.

Leo: Wow. Wow.

Steve: Individual repositories. And it's being downloaded more than 600,000 times every week on NPM for incorporation into those other packages whenever they're rebuilt.

Now, of course, the bigger concern here is the use of what should be a rigorously politically neutral software API being repurposed to inject its author's political sentiment, whether or not we agree with it - I happen to, but still, you know - into the use of their software package. The users - and think about this. The users who receive this sudden antiwar protest pop-up have no idea where it's coming from. They don't know that it was buried in some inter-package API dependency, and that it wasn't put up and reflective of the website or web app they're using. In fact, since everything else they see is coming from the website or web app they're using, that's exactly what they're going to think.

So it really seems wrong. It's the abuse of the implicit trust by the developers who have chosen to use and depend upon this package that's the problem. And over time, with repeated incidents like this this is the third one recently, you know, that we know of, deliberate alteration of the package for this purpose - the abuse of this trust is going to weaken the entire ecosystem. And maybe that's not, in a way, such a bad outcome. Perhaps it should be weakened. That is, perhaps we need to revisit all of this. The very fact that a package's author and maintainer was able to cause their package to behave in a way that its dependent users may well disapprove of should serve as further demonstration of just how rickety, from a security standpoint, this entire package repository, you know, dependency tree ecosystem has become.

In the case of NPM and the browsers that run this code, they're going to start needing to not trust the code that's being sourced by the same-origin server. Not just sequester non-same-origin code, but the stuff coming from the origin server. And if that has to happen, that's a game changer. And of course NPM is only one instance from the world of open source public repository supply chains. There are many more. Maven, Java's similar supply chain, is equally prone. So what we've built is not robust in the face of an active adversary. And unfortunately our adversaries are becoming more active.

I did want to quickly note, for anybody who might be hanging back, that April 2022, next month - oh, no, we're in April now. Sorry, we're in April now. Next month, May, will be end of service life for Windows 10 20H2, and a different form of it for 1909. For Win10 20H2, which was also known as the October 2020 update, it reaches end of service life for Home, Pro, Pro Education, and Pro for Workstations users. The Enterprise, Education, and IoT Enterprise editions receive one additional year of support, so they will be reaching their end of life on May 9th, 2023.

And this also means that next month the already end of service for Win10 1909, which previously ended for Home, Pro, Pro Education, and Pro for Workstations users, will also finally be ending for their Enterprise, Education, and IoT Enterprise editions next month. So next Patch Tuesday, May 10th, will be the last round of updates for anyone still on Windows 10 20H2 who's not using Enterprise, Education, or IoT. And so, you know, that means you get two months, basically, 60 days from now until you would have received an update, which you won't, until you update. So just a heads-up for anybody who may have been holding back and for whatever reason deciding that you wanted to stay where you are. You will stop being able to get security updates.

And just a random little bit of miscellany. We have a neighbor whose son uses Coinbase to manage and retain all of his cryptocurrency, and he's been urging them to buy and hold some bitcoin. Okay. Independent of the value of any cryptocurrency as a buy-and-hold asset - which, Leo, I'm as dubious about as you are.

Leo: Yes.

Steve: I made a comment about the general inadvisability to them of leaving any sizable investment in crypto online, noting that many exchanges had been breached, and that Coinbase had not escaped from that. They suffered a breach back in 2019. And exactly a year ago, between March and May of 2021, they acknowledged that more than 6,000 of their customers were hacked in a large-scale email phishing campaign which tricked their customers into giving up the email addresses, passwords, and phone numbers associated with their accounts. I explained to my neighbors that the only safe practice was to remove any especially large amount of crypto...

Leo: Put it on a hard drive.

Steve: Well...

Leo: And then stick it in the corner of your office.

Steve: Well, and as we know, the only thing you really need to hold onto is your address; right?

Leo: Yeah, yeah. The wallet itself, if you look at the wallet.dat file in many wallets, it's just a long digit, a long number. And that's your account number.

Steve: Right.

Leo: I'll hold his password for him, if he wants.

Steve: So, well, and so I thought it was also interesting that the 6,000 customers a year ago were phished; right?

Leo: Yeah, isn't that interesting, yeah.

Steve: Yes. This discussion made me a bit curious. So I went over to Coinbase.com and attempted to sign in without an account. And I discovered in two seconds...

Leo: Here's the problem.

Steve: ...that they made one of the cardinal mistakes of online security. And, you know, if it was a site where you log in to post about recipes or something, fine. This is Coinbase. They show an attacker when they have guessed wrong about an account's email. So I went to Coinbase, and I put in "bingo-zonk-dingo."

Leo: Ooh, I have to try that in my password for my wallet. That's good. I like it.

Steve: Bingo-zonk-dingo...

Leo: Bingo-zonk-dingo.

Steve: ...@gmail.com.

Leo: And what happened, Steve?

Steve: And I pressed the button, and it lit up in red, and it said: "No Coinbase account exists for this email. Please check your spelling or create an account."

Leo: Oh, that's the worst possible thing they could do.

Steve: Yes, because now it means that any group of attackers or bots can now guess email addresses. And they will be told, you know, it's fine to take the email or account name first and separately. But never tell the user that their account is unknown.

Leo: Because then I could just enter in emails until I get one that says, oh, that's not your password.

Steve: Yes.

Leo: And I know it's an address of an actual user.

Steve: Yes. And now you begin the phishing campaign. So always ask for their password, regardless of whether or not the account is known. Then tell the user that there's a problem logging in. Please check their account name and password. Now, I understand from a customer service standpoint it's much less confusing to a user to provide them with immediate feedback when they've mis-entered their email address or their username at the first stage. But the reason it's much less confusing is exactly why it

provides an advantage to any attacker, who is now able to probe that service for its database of existing accounts. And in the case of Coinbase, an attacker might know someone's email address and wish to know whether they have an account on Coinbase. Coinbase lets them know immediately. I just couldn't believe it. So hacked recently? Yeah. Guess why? Wow.

Okay. Three bits of closing the loop, and then we will do our third sponsor and talk about what's happened with Spring4Shell. Mementh tweeted me, @Mementh, M-E-M-E-N-T-H. He said: "Time-based port knocking. You have an authenticator app, and port knocker gets that and generates the ports to knock." And I thought, that's kind of brilliant. I asked Mementh whether he'd come up with this on his own or may have seen it somewhere since I wanted to give him credit for a brilliant and clever solution.

It solves and resolves the static-knock replay attack and the brute force knock guessing problems in the same way that a one-time password solves the same problems for passwords. Assuming that the client and server are able to both obtain an awareness of the time of day, so they're synchronized, the port listening server could be continuously receiving incoming port knocks into a ring buffer. And whenever it adds a knock to the buffer, it would scan the buffer for the current knock sequence all coming from the same IP address. And if they're present, you're in.

And something else occurred to me as I was writing this up that I hadn't seen anywhere. One nice bit of client-controllable data that's also logged in the typical firewall log, and that's for the implementations where you simply want to be watching the firewalls log and process the log on the fly, most firewalls log not only the source IP, but also the source port. And although ICMP-based knocking doesn't have any port, both UDP and TCP do. So manipulating the knocking packet's source port doubles the number of entropy bits per packet from 16 to 32 without any other additional complexity. You don't have to worry about the content of the packet. It might just be SYN packets; right? And so you can control the source port of the SYN when you're generating it.

So anyway, we got a lot of interesting feedback from people. It turns out that a bunch of our listeners had never heard of port knocking before, and they were grateful for the episode, and also a bunch have implemented it in different ways.

Leo: I like the time-based port knocks. That's a really clever idea.

Steve: Isn't that cool?

Leo: Yeah.

Steve: Yeah. Basically you take the one-time password concept and employ it. I think it's very neat.

Vitrapemli said: "Re port knocking, Episode 865," he said, "I do something a little bit different, but I think it's just as cool. I have iptables log all the packets just above the drop. This is a little messy in the logs, but I have Fail2ban watching the log. If someone is port knocking or scanning my host, three failed attempts on any closed port, I block that IP for a week. The idea is, the people I want to talk to my services on random ports know what port they're on. So they have no reason to try other closed ports." So anyway, I thought that was an interesting approach, too. And I did agree that a mature port knocking system ought to definitely have an IP-based lockout just as an additional layer of security.

And finally, someone who's using the moniker "Lay the proud usurper low."

Leo: Wow. That's Shakespearian. Wow.

Steve: Yeah. That's @ehastings. He said: "Dear Steve. Regarding SpinRite 6.1 and successors, is there any spot on your timeline for a version that is Apple Silicon native?" He said: "Before the new systems I had hoped that there would be a Mac native release. That would seem to be far off, at best. What can you report? Thanks. Gene." And the bad news is no. I can pretty much assert absolutely that I will not be doing an ARM-based version for native Apple Silicon. The next SpinRite will run on Intel Macs for sure. That's definitely on the short-term timeline. But I just, you know, I'm still writing it largely in assembler, and life is too short. It's not fun to hand-code ARM in assembler.

Leo: It would also be really tricky because people who are using Macs almost always are using the Mac Apple controller, which I'm sure is very closed and hidden. I mean, there's USB drives, and there's Thunderbolt drives. You could diagnose those. But they can just put those in a PC. But the drives internal to your Mac, good luck. Good luck.

Steve: Yeah.

Leo: That's not going to be easy. And Steve, by the way, there is no truth to the rumor that he is going to rename his Twitter account "Tyrants fall in every foe." But I think, @ehastings, it was a nice try anyway. On we go with the show.

Steve: So as I noted at the top of the podcast, Spring4Shell is no longer theoretical. Attacks have begun. Last week we introduced the latest Java-based flaw that has been found in VMware's Spring.io web framework, which at the time was still only theoretical. Recall that there had been some questioning even about just how bad this potential RCE (Remote Code Execution) exploit would turn out to be. Flashpoint had said: "Current information suggests in order to exploit the vulnerability, attackers will have to locate and identify web app instances that actually use the DeserializationUtils, something already known by developers to be dangerous." And I was a little skeptical of that, whether developers even know that. I doubt it.

And Rapid7 said that despite the public availability of proof-of-concept exploits: "It's currently unclear which real-world applications use the vulnerable functionality." It's less unclear now. And they also said: "Configuration and JRE version may also be significant factors in exploitability and the likelihood of widespread adoption." I don't know why everybody was, you know, the security firms were downplaying this.

But CERT's Will Dormann tweeted that: "The Spring4Shell exploit in the wild appears to work against the stock 'Handling Form Submission' sample code from Spring.io." And gee, do you think anybody would have taken that sample code and just modified it a little bit for their own purposes? Hmm. I don't know. But maybe it's a reference implementation. "If the sample code is vulnerable," he said, "then I suspect," he tweeted, "that there are indeed real-world apps out there that are vulnerable to remote code execution." And recall that this one got a 9.8 on the CVSS scale.

Now CISA is warning of active exploitation of the critical, it's now considered obviously at 9.8 it's critical, Spring4Shell vulnerability. So it appears that 9.8 was prescient and is

being earned. CISA has added it, the Spring4Shell vulnerability, to its Known Exploited Vulnerabilities Catalog, that's with capital K, Known Exploited Vulnerabilities Catalog, based on "evidence of active exploitation."

Praetorian researchers Anthony Weems and Dallas Kaman noted that: "Exploitation requires an endpoint with DataBinder enabled, in other words an HTTP POST request that decodes data from the request body automatically and depends heavily on the servlet container for the application." Okay, now, the automatic decoding of a POST fits in with Will Dormann's observation that Spring.io's sample code for handling form submission is itself vulnerable. Although details of in-the-wild abuse are still a bit unclear, the information security company SecurityScorecard said: "Active scanning for this vulnerability has been observed coming from the usual suspects like Russian and Chinese IP space." And so I guess Russia's still connected to the Internet.

Anyway, Spring4Shell vulnerability scanning activities have also been spotted by Akamai and Palo Alto Networks' Unit 42, with the attempts leading to the deployment of a web shell for backdoor access and to execute arbitrary commands on the server with a goal of delivering other malware or spreading within the target network. So no big surprise there. Spring4Shell has created yet another new way of jimmying the front door lock in order to install a permanent backdoor. Check Point Research said: "During the first four days after the vulnerability outbreak" - again, first four days. This is why that T-shirt at the top of this show notes is so relevant, Exploit Wednesday. "First four days after the vulnerability outbreak, 16%" - 16% - "of organizations worldwide were impacted by exploitation attempts."

And they added that they had detected 37,000 Spring4Shell-related attacks over the weekend. I have a graph of the explosion of the scanning for this vulnerability in the show notes, showing on March 31st the little tiny bar, maybe 5,000. Then the next day, April 1st, it jumps to 10,000 in that day. On the 2nd looks like it's around 13,000. And on the 3rd it's a little more than 14,000 per day.

Microsoft 365 Defender Threat Intelligence Team chimed in, stating it has been "tracking a low volume of exploit attempts across our cloud services" - that is specifically Microsoft's - "for Spring Cloud and Spring Core vulnerabilities." There are, by the way, a pair of Spring vulnerabilities, both 9.8. And according to statistics released by Sonatype, potentially vulnerable versions of the Spring Framework account for 81% of the total downloads from the Maven Central repository since the issue came to light on March 31st. Let me repeat that. Sonatype-tracked vulnerable versions of the Spring Framework accounted for 81% of the total downloads from Maven Central repository since it came to light at the end of March. So since that time, four out of five of all downloaded were potentially vulnerable.

Cisco, which quickly jumped to investigate its own lineup to determine which of its products might be impacted, confirmed that three of its products are affected: the Cisco Crosswork Optimization Engine, Cisco Crosswork Zero Touch Provisioning, and Cisco Edge Intelligence. All are vulnerable. So if you know if you or your organization is using those, make sure that they're patched because this thing is exploding in terms of bad guys looking to exploit it.

VMware, Spring.io's parent company, has said that three of its products are vulnerable: their Tanzu Application Service for VMs, the Tanzu Operations Manager, and Tanzu Kubernetes Grid Integrated Edition. They've made patches and workarounds available as needed. VMware said: "A malicious actor with network access to an impacted VMware product may exploit this issue to gain full control of the target system." Okay. Saying that the way they would say it if it weren't VMware, anybody on the Internet who is able to access an unpatched VMware instance can gain full control of the target system. So that's not good. So again, look how quickly we moved from "There may be a problem

here, but we're not sure," to "Oh, crap, cancel Christmas." That's the reality of today's world.

To flesh this out a bit further, SecurityScorecard wrote that on Thursday, March 31st, a patch for a widely used Java framework called the Spring Framework was given the designation, and then they list the CVE, it's 22965, with a CVS Score of 9.8. That's the bad news, they said, for a lot of companies that make use of this framework for delivery of their web applications, services, and APIs. They said this is a remote code execution vulnerability, and the ease of exploitation is partly why it has earned a 9.8 out of 10 on the CVSS Score.

And they reminded us that way back in 2010 there was a remote code execution for the Spring Framework v2.5 which fixed the vulnerability discovered then about unsafe class.classLoader.URLs. That was where the problem was. This new remote code execution is related to that vulnerability. The fix 12 years ago was to forbid jumping from a class to classLoader, and the fix this time is to forbid jumping from a class to a module. So basically one step up in the hierarchy. That was, you know we talked last week about the fact that there had been a problem 12 years ago, and that the new exploit was a workaround of that problem.

So the point is this has been present and vulnerable for 12 years. It's just that no one stumbled on it. They were blocked by the change that was made 12 years ago. So let's go up a level in the hierarchy and go in there. So the saving grace is that this only became present in JDK9 and hence. I actually saw some suggestions that if for some reason it was not possible for an enterprise to update their instance of Java, for some reason, they suggested if you recompiled around JDK8, and your app was compatible with JDK8, that was another way of solving the problem since it did not have the bug.

So if anyone's interested in much more detail, the deepest level of nitty-gritty about this was in the SecurityScorecard site. I've got a link in the show notes. But they said in their conclusion, they said: "If this feels all too familiar and is reminding you of the Equifax hack that was due to an exploitation of the Apache Struts 2 framework, then your instinct is spot-on. This is the same kind of vulnerability."

And on top of everything else, the Spring4Shell vulnerability is also now, since the start of April, being actively exploited by threat actors to execute the Mirai botnet malware and for some reason focusing at the moment at least within the Singapore region. In their posting titled "Analyzing the Exploitation of Spring4Shell Vulnerability in Weaponizing and Executing the Mirai Botnet Malware," Trend Micro researchers said that: "The exploitation allows threat actors to download the Mirai sample to the /tmp folder and execute them after making a permission change using chmod."

They wrote that they began seeing malicious activities at the start of April, and they also found the malware file server with other variants of the sample for differing CPU architectures. And of course that makes sense since Java is a multi-architecture language that's executed by its own JVM.

Trend Micro's write-up is by far the most in-depth, even more so than SecurityScorecard, and it's the most detailed analysis that I have encountered. So I've got a link in the show notes for anyone who's interested. And it makes sense that botnets would be quick to jump on this because it's going to be to some degree a time-limited vulnerability. And it's not the first time we've seen this. In December of last year, multiple botnets including Mirai and Kinsing were found to be leveraging the Log4Shell Java vulnerability to breach susceptible servers on the Internet. And as we know, Mirai, which means "future" in Japanese, is the name given to the Linux-hosted malware which has continued to target networked smart home devices, you know, IP cameras, routers, and then link them into botnets primarily for DDoSing.

Intel 471 researchers said last month that: "The Mirai code is so influential that even some of the malware offshoots are starting to have their own code versions released and co-opted by other cybercriminals." Remember that Mirai source code escaped and was then found in the wild, and some other offshoots of Mirai were created. In January, CrowdStrike noted that compared to 2020, malware targeting Linux-based systems had increased by 35% during 2021. Intel said that: "The primary purpose of these malware families is to compromise vulnerable Internet-connected devices, amass them into botnets, and use them to perform distributed denial-of-service attacks." And of course we all know all too well just how powerful and prevalent DDoS attacks have become today.

So anyway, once again we see, you know, it was the first item that we talked about last week was that someone had discovered a new way around a problem that had been patched 12 years before. The security community was like, well, we're not sure. Maybe it depends upon the settings. But Will Dormann said, you know, the default sample code is vulnerable. What do you know? Yeah. The reference implementation, it's vulnerable. You think that might be a problem? Uh-huh.

Leo: Wow. Wow.

Steve: Wow. So, yikes, 37,000 compromised attacks at this point.

Leo: You kind of feel like you're hearing news happen right in front of your eyes. You know? It's kind of amazing.

Steve: Yeah. Yeah.

Leo: Yeah. It's why it's worth listening to Security Now! every Tuesday. We do it around 1:30 Pacific, 4:30 Eastern, 20:30 UTC, if you want to watch or listen live, at live.twit.tv. If you're watching live, chat live at irc.twit.tv or join our Club TWiT. You can chat in the Discord. Actually that's just a small fraction of the things that happen in the Discord. It's a very active place to go to talk about all kinds of geeky subjects. And you get ad-free versions of all the shows, and you get the TWiT+ feed, which is full of stuff that didn't get on-air, or shows that we're preparing for a future in the public, like the untitled Linux show and Stacey's Book Club, and This Week in Space recently came out of the TWiT Club, is now public. All of that for seven bucks a month at TWiT.tv/clubtwit.

You can also get copies of the show after the fact from Steve. He's got two unique versions of the show, a 16Kb audio version for the bandwidth-impaired. He also has beautifully crafted, human-crafted versions of the transcript at his site: GRC.com. While you're there pick up a copy of SpinRite. That's his daily bread, the world's finest mass storage maintenance and recovery utility. 6.0 is the current version, soon to be 6.1. You'll get 6.1 for free if you buy today. But you'll also get to participate in the development of it. That's at GRC.com, along with ShieldsUP! and all of this free stuff and lots of good information: GRC.com. You can leave Steve feedback there at GRC.com/feedback. But it's even easier to do it on his Twitter account. He's @SGgrc, for Steve Gibson, GRC, @SGgrc. Steve Gibson, Gibson Research Corporation, on the Twitter. And his DMs are open. So slide on in. Leave him a message.

We have copies of the show, 64Kb audio and video, at our website, TWiT.tv/sn for Security Now! There's a dedicated YouTube channel, of course, as there is for all of our shows. Best way to do it, though, as with any podcast, is get a podcast client, there are very many, and just subscribe to Security Now!. That way you get it automatically. You don't have to think about it. You just know it's there of a Tuesday, ready for your listening.

Steve, have a great week. I'm going to go back to v2 of the Bobiverse, or Volume 2.

Steve: Oh, good, yeah.

Leo: Catching up with you.

Steve: I'm continuing to wade through #4.

Leo: Four, yeah.

Steve: I'm at like 83%, and it's like, okay, well, I have to finish this, but...

Leo: There's no #5; right? Four is the last one?

Steve: Yeah, there's not. He's actually talking about, threatening, I should say, a fifth one.

Leo: Okay.

Steve: But he's busy doing some other stuff.

Leo: Okay. But the first, at least I can vouch for the first two. They're great, yeah.

Steve: Oh, Leo, it is definitely fun. The trilogy is worthwhile.

Copyright (c) 2014 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details: http://creativecommons.org/licenses/by-nc-sa/2.5/