POST-MORTEM OF A MAINFRAME HACK

In this episode of BMC AMI Z Talk, Grant McDonald and Chad Rikansrud return to the show to share examples and cautionary tales of what happens if you fail to take proactive steps to safeguard your mainframe. Below is a condensed transcript of our conversation.

Grant: So, Chad, I know obviously, you can't give out any names. We have to protect the names of those that are involved here, but what are the couple, you know, one or two more interesting cases maybe that you can think of where something you noticed when you were performing a penetration test that kind of surprised you?

Chad: Hey, Grant. Well, thanks for having me back. Yeah, I'll tell you. Two things that surprise me continuously when I'm out looking at client sites – and you know, just for listeners, to remind them, these are some of the biggest, most complex organizations on earth, right? I mean, these are the biggest financial institutions or government or healthcare companies or retail companies or airlines, whatever.

I mean, when I talk about this kind of stuff, what kind of adds an exclamation point or some gravity behind it is just what's at stake with these systems because like I said last time, and I usually say this every time I give a presentation, is if you're running a mainframe, you're probably doing something important with it, right? Or put another way, if it isn't working correctly or if it gets compromised, your business isn't doing business, really operating in any kind of meaningful way. And so, two things that I – and they're big things, but kind of broad-level things that I see that are just generally shocking.

One is just the basics. There are – and we can talk about this later. I've talked about this in a lot of other different podcasts and presentations and talks and stuff. But there are so many basics on the system with respect to security that just don't get done.

I mean, for people out there that are mainframe fluent, things like having update access to APF-authorized libraries or things like the number of people that have elevated privileges on their own IDs that still exist or the number of network connections that require no authentication, whether it be anonymous FTP or a JES that doesn't require any – like network job entry that doesn't require any usernames, passwords, or keys, anything like that.

Those things still exist in the wild, and they're kind of shocking because all of that tells me that the folks sitting behind those systems are really relying on their perimeter security in the hope that none of their internal users that have access to the system over the network are ever compromised. So, that's No. 1, and I just mentioned a couple of them, but there are many of those that fall into that sort of basics bucket. Before we even get to things like No. 2, which is just kind of shocking to me, is the number of times we find bona fide back doors on the system. This is something that on any other platform anywhere else would be just unheard of or crazy.

I remember – it's been several years now, but I remember there were a couple of routing and switch companies that I won't mention by name, but where they found backdoors in the code that it looked like they had been there for years and don't really know where those came from. You know, there's always speculation about government actors or foreign actors, that kind of stuff.

But what we have found while doing assessments of clients' systems are everything from software vendors that have intentionally put back doors on systems so that they can better support them, meaning that they can come in – if they get access to your system, they can give themselves whatever permissions they need without having to ask you, which is just absolutely crazy. I mean, it would be like someone installing an alarm system in your house and making sure that they had access to disarm it at any point in time, even if the alarm went off or whatever. So, that's kind of bananas. I'm actually going to talk about one of the more egregious ones that I've ever found.

And the other part of it is we see – sometimes, we see users of the systems who probably had their privileges taken away through some enterprise initiative where they're removing privileges from people's daily driver's ID, which is the right thing to do, and they sort of, you know, take that not lying down, kicking and screaming, and as a result, they add themselves a little magic backdoor, a little magic system call or supervisor call or program call that lets them basically escalate their own privileges if they need to.

And I tend to think people do this with – if there is sort of a good intentions, with good intentions, but it's terribly shortsighted. If I can find this in a pen test where I've been looking at the system for a couple of hours, then you know, a bad person who is working on your network that has a lot of time is certainly going to be able to find it as well.

Grant: Gotcha. And in most of these incidents, do you think there was something that there should've been an alarm sounded for somebody that they just didn't notice, or do you think they were just completely unaware that this kind of thing was happening?

Chad: I think up until very recently, in terms of – in mainframe terms, right? Because we talk about the history of Z, we're talking about 50, 60 years, right? So, up until very recently, in the last few years, there hasn't really been a good way to find these things that was commercially available. Now, there's a couple products out there that can help you find these things, maybe, right? Some of them, not as much. But you know, I think that they didn't – to answer your question, I think that there wasn't any way of sort of proactively finding these things, or even after the fact, without looking at it.

And generally speaking, you have to trust the systems programmers or the security folks. They have the keys to the kingdom, and there's very few people in the enterprise with the knowledge to check it, that would know how to check what they had done or what they're doing, and if your system's not locked down to begin with, it's pretty easy to install what we call in pen testing some type of persistence, which is just a fancy way of saying a back door that lets you get back onto the system or lets you get back to escalating your privileges. Pretty easy to do that, and a lot of systems programmers or whatever would know how to do that, and we certainly have found those in multiple systems.

Grant: Yeah, I'm curious. Obviously, part of your day job is actually performing penetration tests and to try to act and think like a bad guy and see if you can get into somebody's mainframe. When you're out there, do you often have somebody who works in either operations or security coming and saying, "Hey, what are you doing? I saw you pop up over here or over there." I mean, does that already happen very often, or does it generally happen too late in the cycle at a point where you could've effectively, if you were a bad guy, have taken over control?

Chad: That is a great, great question because a lot of people maybe don't think about that one of the reasons we pen test is obviously to find vulnerabilities in the systems, missed configurations, missing patches, another back door, like I've been talking about. Those kinds of things. Another reason, though, we do this is to test out the responses, so sometimes if it's more of a red team exercise or even just during a pen test, we may do it where the SOC, for instance, doesn't know we're doing it, intentionally, because we want to know what are they able to detect.

And honestly, of all the times I've done this, only once have I had somebody come crashing through the door and tell me that they saw something that I did and that sort of thing. And even then, it was because I had moved past the period where I was trying to be very stealthy to being very loud. So, it was the equivalent of being a cat burglar and tiptoeing around the house to when you realize nobody's home and you just start flipping on lights and turning on the radio and checking out the fridge to see what there is to eat while you steal things. And then, they actually came out and said something about that, which was better than nothing.

But most of the time, funnily enough, the way we get noticed is just because we might be running a job that's sort of searching data sets for passwords or crypto keys or things like that, and those are kind of brute force things you do, and sometimes they take up CPU, and so often, we'll get – if we get outed, we'll get outed by operations, who's called us in the past and said, "Hey, that job you're running is taking a lot of CPU. Is it okay if I knock down the priority a little bit?" We're just like, "Yeah, sure. That's fine."

But they're not really asking, "What are you doing?" They're more just concerned that you're taking up precious CPU cycles, which interestingly enough, there have been – you know, I can't name names, but there have been very high profile breaches where they weren't mainframe breaches, but the mainframe was delivering the data behind the scenes, so imagine a website getting breached and people figuring out how to query databases to dump data from some company where the mainframe is the back-end database on that, so it's not a mainframe breach, but that's where the data's coming from.

And the way that they actually found the breach was because of the number of CPU cycles and number of jobs being spawned on the mainframe. So, it's a poor detector, but it's one of the things that mainframe operation shops are conditioned to look at because historically, that's how you keep a good – a smooth running shop is by making sure you don't overuse your CPU or have too many jobs running at the same time or you're not running up the credit card on the system.

Grant: Interesting. And I wonder, when you perform those tests, I think some people think that you're starting from basically a high-ground position where you've already got mostly what you need to compromise, and of course, you'd be able to do it. But is it something where you start from a point where you've got some level of access that you're able to leverage, or is it more of an uphill battle when you do a penetration test?

To listen to the rest of this episode, visit SoundCloud or Apple Podcasts