EPISODE 1185

[INTRODUCTION]

[00:00:00] JM: Network discovery allows enterprises to identify what devices are on their network. These devices can include smartphones, servers, desktop computers and tablets. Being able to index the devices on a network is crucial to figuring out the security profile of that network. HD Moore is the founder of Rumble Networks, a company focused on network discovery and asset inventory. He joins the show to talk about how network discovery works and his experience building Rumble.

[INTERVIEW]

[00:00:35] JM: HD Moore, welcome to the show.

[00:00:36] HDM: Thanks, Jeff.

[00:00:37] JM: I'd like to start off by talking about a modern IoT network. So when I say IoT network, I kind of imagined this like network of sensors and stuff in a factory, but I don't really know what goes into an IoT network. Can you give me an outline of what an IoT network actually entails?

[00:00:57] HDM: Oh sure. There're lots of varieties of it. You have kind of your typical consumer devices which are now showing up more and more in corporate networks as well, but you also have all the fun operational technology stuff, things like clinical equipment within healthcare, energy and gas, serial port servers, terminal servers, various monitors, sensors, things like that. So it's a really wide gamut for IoT in general. The things that most consumers are familiar with tend to be their talking voice assistants, their smart TVs, their Nest cameras, things like that. On the industrial side it really is pipeline sensors, pressure gauges, thermostats, industrial cameras, things like that as well.

[00:01:31] JM: What's hard to secure about IoT networks?

[00:01:35] HDM: The biggest challenge with most IoT devices and security really comes around to updates and management. These devices are generally built to be fairly rugged. They generally don't have a lot of inputs or user interfaces on them. So there's not really a good way to manage them. As a result, most IoT devices either auto update or they don't update at all and they require some heavy manual work to be able to apply updates to them. The result there is that when there is a security problem, when there is an update that's needed, when a configuration change is required, it's either really, really intrusive to figure out how to actually fix them. You have to physically reset them or attach a serial port, things like that, or you can only manage them through a vendor portal.

[00:02:10] JM: So you started Rumble Networks. What did you start the company to solve?

[00:02:16] HDM: My background really comes out of like pen testing and security assessments and breaking into stuff, and the very first thing we ran into with almost any security engagement and any security tool I worked on previously, Metasploit or something else, was discovery. It's like what's actually on the network? There're lots of great discovery tools today, but they're really good at telling you what software's running on these devices. They're not really good at telling you what those devices are. So the goal behind Rumble was how do we build something that quickly tells you what's on the network? What type of devices they are? What the vendors are? All that fun stuff.

[00:02:44] JM: Can you tell me a few things about the internet of things that might surprise me like in terms of building security products for it? In terms of building asset inventory products for it, what's difficult?

[00:02:57] HDM: Probably the biggest gap between your standard IT equipment and the IoT devices is really that manageability layer. Any devices either have a really weak and fairly insecure management interfaces. Think about the web interface on your Linksys router , that

kind of thing, or they're purely managed in the cloud and they expose almost nothing to the network itself.

So, for example, if you scan a Nest camera with Nmap or another discovery tool, it's not going to tell you pretty much anything. It's going to tell you maybe the MAC address if you're lucky and that's pretty much it. They really expose no services to the local network. Flipside though, things like Philips Hue light bulbs, they actually have a full-blown web server. They've got management interfaces. They've got APIs. All kinds of stuff exposed directly to the network itself. And you have to query them a little bit differently than you would a typical device to figure out what they are. How out of date they are? What network they're associated with? Things like that. So you end up having to build a lot of very specific per device fingerprinting and per device discovery techniques to be able to accurately inventory IoT devices.

[00:03:51] JM: You created the Metasploit project. Can you explain what Metasploit is?

[00:03:55] HDM: Yeah sure. Back in 2003, the world was quite a different place. There wasn't a lot of like public exploits. I was working at the time as a consultant and we needed exploits. We needed tools to let us break into networks, and there weren't really a lot of great tools. Most of the tools that we were using were kind of coming out of the underground and from the community, but there wasn't really like a formal kit you could pick up. So Metasploit became a way to kind of standardize exploit development and also a way to kind of take all the really cool research coming out of security community and wrap it up in code and then maintain that code going forward. So it's turned into this massive project. I think there's probably hundreds of thousands of users at this point. The product's been running for almost like 16 or 17 years. I ran the project for about 15 years straight in various roles both community open source and then eventually into Rapid7.And it's a lot of fun. It's basically building evil in an open source kind of aquarium and letting anyone contribute to it.

[00:04:46] JM: Is it a sort of registry for vulnerabilities that automatically tests vulnerabilities across a network?

[00:04:55] HDM: A little bit. Really, the core of Metasploit was giving you a set of modules. So a module is kind of an individual type of vulnerability test you can do. But compared to something like Nexus or Nexpose or KQualys where they just tell you that it's vulnerable. Metasploit actually lets you break into the system and actually gives you a shell or gives you a screen or gives you a remote control of the device.

So the whole goal of Metasploit is to let you actually break into something. So part of it is testing the vulnerability, but the other half of that is now that you've tested it, know that it's vulnerable, how do you actually do something useful with the device now that you've broken into it?

[00:05:25] JM: And does that posture of designing a security tool of actually allowing you to break in rather than just detect it, does that allow for misuse of the security product?

[00:05:35] HDM: Oh, of course. Metasploit was very controversial early on. These days people kind of take it for granted that that's a thing that you can do. I think early on folks were trying to get me fired, trying to get me arrested, all kinds of fun stuff. So a lot of folks really, really didn't like Metasploit. Even today a lot of folks think it's not a good thing to do.

The flip side though is without having open source tools out there that demonstrate these vulnerabilities and do it for real, not just pretend, it's really hard to build defenses. It's really hard to test. It's really hard to train people. And for folks who do consulting work where you actually need to verify these exploits by actually trying to break into stuff, you actually need these tools. There's not really a good way to build these tools in a way that only the good guys get to use them.

[00:06:12] JM: Why is that? Like why is it so important to build a security tool that allows you actually access the vulnerability as opposed to just detect it?

[00:06:21] HDM: A lot of it comes down to kind of defense and depth. Like if you're doing a good job of securing your network, it doesn't matter whether you've got a single vulnerability or

a single exposure. It matters whether if someone exploits that, can they actually get into it? Can you detect them? Can you detect the moving around afterwards? There're a lot of other parts that are involved besides just that initial vector.

And so determining that the vulnerability is there is part of that. But verifying that all your other defenses in depth not only detect but can prevent expansion from that exploitation, that's a bigger part of it. And if you're doing a red team assessment where your job is to look for solutions and find some way to break into a network and then use that to gain access and move around, you really need a tool that actually gives you real access during that first step. So the good news about having all those tools in kind of the open source world and kind of in the public domain is that if you're building defense tools, you can then use Metasploit to build better defenses as well.

[00:07:13] JM: How did your work on Metasploit inform what you've built with Rumble?

[00:07:17] HDM: One of the things I really enjoyed about working on Metasploit was finding really neat information leaks. So if you're trying to exploit a vulnerability and it's like a memory corruption bug, you generally only have one chance. You have to get it exactly right with all the right magic values. Otherwise you crash the service as opposed to making it crash in the magic way that gives you remote access to it. So that's kind of the magic behind exploits. And a lot of that is really building up on what we call information leaks. It's finding really kind of benign ways to leak really small bits of information that you kind of build them all together and create bigger picture of what that system looks like so you can accurately exploit it.

And that information link piece was really fascinating to me. I really enjoyed finding ways to remotely identify bits about a system or a device without causing it to crash just kind benignly through network traffic. So one of the cool things about Rumble is we got to take a lot of the techniques that were originally developed for security testing and for pen testing that weren't exploit parts. They're just the discovery parts, and bring that into kind of a full-blown commercial platform for asset inventory and network discovery.

[00:08:13] JM: So the goal of Rumble is to have a tool that can inventory all the devices and the ports on a network, but you don't need credentials, right?

[00:08:26] HDM: That's right. Most of the tools IT folks use today really depend on having like actual directory credentials or SSH credentials to go enumerate things. Same thing on the security side, a lot of folks are using their vulnerability scanning tools as their inventory today because that's the tool they have. What we're trying to do with Rumble is really focus just on asset inventory, just on network discovery. So you can drop an agent in, you can run some scans and you quickly identify everything attached to your network with no credentials. Whether you have access network or not, you can quickly see what's there. So it's really useful when you're acquiring another company and you're merging your networks together. When you're consulting going into a brand new network you haven't seen before. If you're the new IT guy and you just took a new job and you're walking into the work the first day and you want to figure what's out there, you've got your spreadsheets, you got your CMDB, but nothing's better than kind of a real view of what's actually connected.

[00:09:08] JM: And how do you map what's going on in a network without credentials?

[00:09:14] HDM: It's tricky and it's fun. I love this part of it. So the challenge really is that because you can't log into individual assets, you have to find ways to leak information about each asset that helps you uniquely identify it over time. So, for example, Rumble has like 15 different ways to obtain a device's MAC address remotely. If you're on the same layer 2 network, you can also use ARP. But if you don't have that, there's a billion other ways you can get there. You can pull it out of net bias responses. You can pull it out of UPNP replies. You can decode it out of TLS certificates. There're all these other neat techniques you can use. And the MAC address itself is just one part of the revolve device fingerprint. You can also look at things like the unique GUID that comes back from the SMB protocol 2, or SMB version 2 protocol. There's lots of other kind of hints and bits that come back as well, like the TLS certificates associated with remote desktop ports.

So what Rumble does is tries to identify all these different unique attributes and all these different little small information leaks and then put those together into a device profile and then turn that into a fingerprint and say, “Hey, this isn't just a device running Linux 2618. This is a printer. This is the desktop. This is a IoT device or a Philips Hue light bulb.”

[00:10:13] JM: Gotcha. So when Rumble starts to index a network, what happens? Give me kind of the run cycle.

[00:10:22] HDM: Oh, sure thing. Yeah. So the idea is we want to send the least amount of traffic you possibly can to uniquely identify each asset. So we're not trying to replace your vulnerability scanner. We're trying to tell you what the devices actually are, whether it's a desktop or a switch or something else.

So the very first steps are things like an ICMP ping where you send a boring old ping packet, but even the way you do that's a little bit interesting. We'll actually encode the destination of each ICMP ping is encoded into the payload with a magic decryption key and then when we get a reflection, we get a response back from the device, we then decrypt that and verify that.

So the majority of the Rumble scanning engine is stateless and that we're embedding metadata into the packets themselves so they reflect back from the targets, we know where they came from. The neat thing about that is let's say we send ICMP ping messages to our entire /24. If we get a response back for two devices and the destination was the same for both, we know that device is multi-homed. So we can actually identify multi-home devices through techniques like that where by embedding unique identifiers into our pro packets, when they reflect back, even though it's just benign, boring traffic, we're able to decode those, verify those and then peel out the original destination and then tell you more about the network as a result.

[00:11:26] JM: Very interesting. So give me some of the high-level engineering problems you've tried to solve in building Rumble.

[00:11:33] HDM: Oh sure. Let's see, there're lots of fun ones. MAC addresses are a fun one, because MAC addresses are how a lot of organizations uniquely identify their assets. If you can't directly measure a MAC address through a protocol like ARP, what do you do? So we'll do a UDP net bias request. We'll get the MAC address out of that. Let's say you're scanning a network multiple hops way, maybe the remote site, and all the desktops and the network have a firewall label. And so you can't query them directly. So how do you get the MAC address instead? Well, we've got a couple techniques. If you have credentials like an SMB version 2 public string or version 3 read-only credential for SNMP, you can query the switches cam table and dump the MAC addresses from that. If you don't have that though, maybe there's a printer on that network and they can query the ARP cache of the printer itself through SNMP and that'll tell you the MAC addresses of those devices.

Now the great thing is once you have the MAC address, you also may have other information about the device. Okay, if that MAC addresses was only issued in the last six months, you know it's a fairly new modern device. And depending on the vendor, depending on the prefix and the range, we're able to tell you that not only is this thing a recently manufactured device, but it's an Amazon Echo or it's a surface tablet or something like that. So there're lots of really interesting challenges that happen in that layer.

A second component is even once we have those unique identifiers for a given asset, oftentimes devices lie about them. For example, a lot of FortiGate firewalls will send bogus responses back for every IP address. Certain knack devices will send the same MAC address back for every single IP no matter which one you talk to. Our proxies and wireless access points do the same thing.

So a large portion of what we do in Rumble is trying to figure out did the response we get back, is it accurate and is it lying to us? And how do we make sure that it's not a bogus response? And then uniquely track that device as it changes over time even without having your standard authenticated access to it. So lots of really fun engineering challenges with tracking devices, with reducing bogus replies, with normalizing results, but picking the best fingerprint.

[00:13:20] JM: That strategy of doing a sort of reverse engineering of what is on the network, is that problematic in terms of getting comprehensive network coverage? If you're just doing these like hack after hack after hack of finding where the printers are, finding where the Amazon Echos are, can you actually get comprehensive network coverage?

[00:13:41] HDM: That was the bet. I mean early on I wasn't really clear whether we can get there. These days we're tracking three million assets per day in the platform. We've got over 100 customers using the system. And yeah, actually like you have to get a lot of different data points. One of the really cool things about where we are today with Rumble platform is that we have so many different fingerprints for a given type of device that when we add a new fingerprint we can automatically generate fingerprints for new fingerprint types now. So because we're measuring the device three or four different ways already and we're getting lots of different hints about what type of device it is, then our analysis section will say, “Hey, three things say it's this. One thing says it's that.” When we add that fourth or fifth fingerprint to it, we can often synthesize fingerprints automatically based on existing fingerprint coverage. So when we add like a new detection technique, oftentimes we can automatically create a huge bucket of those fingerprints at once.

Like, for example, when we created a fingerprint for favorite icons, those little interesting icons that show up in every website you visit. We pull those automatically off the device. We show them to the user and we actually were able to mass generate the majority of our fingerprints for those by using existing fingerprints we had for those platforms and those devices.

[00:14:41] JM: And are there any – Like how do you even know where the weak points are in your strategy? You got the strategy of deducing what the heck is going on across a network. There must be some weak points in it. Or can you even know? I guess if you knew, you would figure out a way to patch it up. But do you know if there are weak points? Can you give any guarantees about the percentage coverage of your inventory solution?

[00:15:09] HDM: Oh, sure. It really depends on the type of network you're looking at. For networks that are your standard corporate environment, we get pretty close to 95%+. The reason for that is almost every Windows device has the NetBIOS service enabled, has an SB2 service or a remote desktop, and that's usually enough for us to fingerprint it. All other network devices, whether it's like Cisco switches or NETGEAR or whatnot, we can fingerprint those pretty easily. Just about all the major vendors we've got pretty good coverage for today.

Where that starts to tail off is when you start getting into more consumer IoT devices or down really deep vertical holes, like medical device equipment, some energy and transmission equipment that we haven't seen before. But our process for that is what stuff is on the network that has at least one identifiable service that we weren't able to identify? So the great thing about being a cloud-first product where you really just install one quick agent and everything else is hosted for you is that the back end's constantly getting better. Each time we run a scan and we find something that we don't know what it is, that goes into our queue and we work in making it better every time.

[00:16:04] JM: I get it. So, frequently, you can find devices where you know you don't know what it is?

[00:16:14] HDM: That's right. And some of them we can't do anything about. So, for example, if it's an Amazon Echo but it's not on the same local network and we can't get the MAC address, there's really no ports open the device by default. Same with Nest cameras, they're probably the worst and the hardest. They may respond to ICMP, but there's very little else you can get from them remotely.

So there's a lot of research that happens on our side. Like how can we tell these two things apart? How can we figure out what the vendor is of device even if we can't get a unique reply through some other mechanism? And that's the part I really enjoy though, is really just getting deep in the weeds with Wireshark and trying to figure out how to fingerprint something.

[00:16:44] JM: And what is that process like of tinkering and tinkering and tinkering until you find the key to identifying a device?

[00:16:52] HDM: It's fun, but it's tough enough for the faint of heart either. I mean I think our favorite icon database work even though we had a lot of hints to start from probably still took about 20 hours straight of us going through looking at tiny, tiny little icons, 16 by 16 images, and then trying to figure out what the heck they were. So I think it also applies for you know really low level protocol stuff. We're doing some SMB 2 research a while back trying to identify the difference between Windows and server variants of the same version of Windows.

For example, the windows 10 and server 2019 have the exact same fingerprint bit for bit on SMB 2 all the way up to everything pre-authentication. There's almost no way to tell them apart using your standard SMB 2 negotiation. So that one was really fun as we had to dig really deep into the protocol. And what popped out actually wasn't a way to fingerprint the protocol itself or the device. It’s a way to identify how busy the device was. We found out that there's actually an information leak where the session ID and the response from SMB 2 tells you how busy the server is. And we found some kind of weird crypto bugs there happening as well that Microsoft eventually fixed.

So a lot of stuff comes out of that research that's unexpected and sometimes it's useful for fingerprinting. Sometimes it’s useful for activity monitoring. We can tell you whether a Windows device is being idle or not idle as a result now from doing that work even though the fingerprinting work itself didn't really go anywhere. So it's always kind of a crap shoot to figure out what differences between devices are out there and what you can learn from them, but that's also the reason why I love my job.

[00:18:11] JM: Is it hard to hire for this kind of work? It seems like it's such a like a difficult line of work to be able to do this kind of reverse engineering. Is it just you? Are you the only person at the company?

[00:18:24] HDM: There's two of us on board right now and we've got a third starting in January and a fourth tenably queued up for March. So it's definitely a small company. And you're right. It's hard to hire folks you can do like low level TCP/IP and application protocol stuff. Just like security, you tend to hire people from adjacent fields. You tend to hire developers or engineers or security folks who really like doing like low-level wire analysis. People who are coming out of a SOC work to do incident response or malware analysis. So it's kind of a wide range of stuff.

Myself and the engineer that's currently on board, we're both kind of journalists. We do a little bit of everything and we're both just creative folks who really like to dig deep in this stuff. But it's definitely challenging and you have to really enjoy the low level bits to get value from it.

[00:19:04] JM: What programming languages do you use for Rumble?

[00:19:08] HDM: Almost 100% Go at this point. We have a little bit of JavaScript frontend work that's mostly just boring JQuery and HTML, but everything else is top to bottom Go. We use the Buffalo framework on the website, which is a little hit or miss, but it's been pretty good for so far with some customization. All of our agents are cross-built on a single windows machine right now. We can build pretty much anything. We build for a ton of different platforms, OpenBDS, FreeBSD, Windows, Mac, etc., all from a single system, which is one of the great benefits of Go. Our binary size is a little bit big. So it's about 20 megabytes for an agent, about 20 megs per scanner. But the entire platform that runs everything on the backend of Rumble itself is like 150 megabyte binary that we just kind of build and then deploy as many instances of it as we need. So it's been nice having a nice clean stack where it's not like a thousand different tools and a billion micro services. It really is just one big binary running majority of the platform today.

[00:19:56] JM: What are the advantages of using Go for building a security product like this?

[00:20:01] HDM: For us, because we have to support agents on just about every platform known to man, we need to build a cross compile really easily and we need to have a really light footprint. So having a single style compiled Go binary with no cgo links, no DLLs, no library

links, that was pretty difficult to to get where we wanted to get to. Like we couldn't use LIBPCAP, for example. So we do like raw BPF work. We do raw device and filter compilations and stuff like that to make it all work as a result. But the upside of that is that we don't have any C library dependencies. We can ship on pretty much anything really quickly.

So being able to cross compile for multiple platforms to architecture from a single system to support all of our customers different platforms and OSs out there at once, that was really valuable. And for me it really is simplicity. Like I spent years shipping products that use Pearl and Ruby as the backend, aand the headache that's involved with just getting those products up and running, installing write dependencies, it's not worth it. So I really love Go just for the simplicity. Once you build the thing, it probably works.

[00:20:55] JM: There are a lot of companies that do different kinds of asset device and asset management, and I kind of wonder who operates these things. Like I do interviews with these people who are running the enterprise technology company, like you, or – Oh! There was another one. I can't – I mean I've done a couple interviews with these cloud asset management companies, which is a little bit different. Significantly different than what you're doing. They're like identifying the virtual servers that are idling and not doing anything and costing you thousands of dollars every year. But I kind of wonder who is operating these things at a company. Is there some dedicated DevSecOps person who's using these things?

[00:21:37] HDM: That's a good question. I think when we started the company we weren't really sure what the market looked like, because your IT folks tend to use your standard network IT management discovery products, but they're really designed not for discovery, but for performance monitoring, for failure detection and for making sure your network switches are being overloaded, that kind of stuff. So there's a whole bunch of tools that are just for network operations IT management, which are really about like software inventory and tracking health and systems and detecting failed hard drives and that kind of stuff. That's all great. Then you have the security crew and the security folks generally didn't have an inventory solution themselves. They were relying on either the IT team giving them a spreadsheet or getting access to an IT tool or using their vulnerability scanning platform and their vulnerability list

basically as their inventory. So that was really the challenge, is how do you provide a tool that provides inventory to all the different teams, whether it's network operations, dev, engineering or security and it's still useful?

So what you typically see today is that someone, especially with large organizations, they'll use something like ServiceNow as the backend CNDB and they'll have a bunch of different tools all feeding data into it, whether it's the ServiceNow Discovery engine, whether it's Landsweeper, whether it's some other IT tool like SolarWinds. Those are all basically feeding data into the CNDB. But typically those are authenticated scans. So the only thing they can see is stuff that they can actually authenticate into, which means most the IoT systems out there and most networks that are more than a hop or two away, new subsidiaries that are added to the network, they really have limited visibility into.

And on the flip side the security teams really had not much to go on. They had a list of vulnerabilities, but not really a list of assets. So we've been trying to kind of fit that gap between IoT and security in that sense and provide better visibility really to like the director of security, head of IT, kind of in that bucket. You typically don't see a lot of DevOps folks care too much about inventory mostly because they assume that whatever their configuration management tool is, whether it's Terraform or CloudFormation, that is their inventory. Like they assume that their infrastructure is code. Therefore their code is their infrastructure. And oftentimes they're surprised with that. You tend to find things that are somewhat unexpected.

Like we have a lot of Rumble deployments where the agents are actually running in AWS VPCs. We find all kinds of crazy stuff in those VPCs that the engineers who are deploying instances into them didn't realize were there. Like, for example, when you deploy a load balancer in AWS, it's actually putting a temporary instance into your VPC. Same thing with your RDS instances, things like that. They're actually deploying VMs basically into your private network kind of firmly and constantly cycling them out. So the contents of those networks is actually really surprising to a lot of folks who tend to manage their infrastructure as code.

[00:24:02] JM: What's the remediation path? Or is there remediation? Is it even remediation? Is it just like I run this thing, I find there's like a bunch of random security cameras and – I don't know. Errant servers on my network. What do I do? Am I doing anything? Do I just leave them there?

[00:24:24] HDM: Yeah. It's usually like kind of the first three steps we see folks use of the platform today are, one, kind of initial visibility. Like what the heck's on my network? What type of devices? Do we have do I have a bunch of Huawei cameras that shouldn't be there from a federal contractor? Do I have a bunch of you know printers that are authentic that have default password still? Things like that.

The second phase of that it really is, okay, I kind of know what's on the network. I know which ones to get rid of or to create tickets for and otherwise manage and assign orders to. How do I dig into it and figure out what stuff doesn't meet my policy? Like what devices have SMB version one enabled? What systems don't have SMP signing? What systems have default SMB community string? So it's kind of looking for badness, looking for anomalies and kind of just doing quick hits to resolve those as you go.

Kind of the last piece of that is that when there's an instant, when there's a problem, Rumble has this historical inventory of everything that was on your network going back however long you've been using it. So you can go query for and say, "When did this device show up? How long has it been there? When did it arrive? What ports are open? When did it change IPs last?" Like having all the information kind of the tip your fingers to dig into it and go work with that is really helpful for teams that are doing incident response or otherwise trying to mitigate an issue where they really weren't sure what changed when.

[00:25:28] JM: And if these kinds of things are unmitigated, what exactly is the security vector? Is it just typically like if one of these devices is unsecured, it becomes easier to penetrate the network as a whole and jump between devices?

[00:25:47] HDM: For the most part we just focus on inventory. We don't do a lot of – Even though I come from a security background. The main goal of Rumble is to provide kind of a fact database. A system you can go query and pull things out of. So we don't have a lot of like – There's no like alerting in the platform that isn't about security issues. We're not telling you whether something is good or bad in that sense. We're just saying like, “Here, you have this thing.” And it's helpful in some cases for – If you're trying to figure out when a malicious device joined the network or someone puts an access point with an open SSID and connects that to your internal network, we'll find all that stuff pretty quickly. But it's really kind of search driven. You have to go look for it in that sense. If you're trying to find TLS certificates that are about to expire or TLS issuers that are signed by an authority that was revoked for some reason. Those are all things you can use the Rumble back and to answer.

[00:26:30] JM: So tell me about building this fact database and updating it as a whole. I guess, first of all, like what kind of backing storage do you want? Do you just use like a SQL server and kind of what happens on each scan of the network? How does the database get updated?

[00:26:49] HDM: Oh, yeah. So when you run a new scan, it's going to send things like you know ARP request, ICMP, TCP scans, banner grabbing, all that fun stuff. That all turns to kind of a raw dataset. That's just a list of this IP editor has had this probe response and that's it. Our backend takes all that stuff and merge it together first by IP editors to group for that scan. And then it tries to group IPS together based on which IPs are actually part of the same asset.

So, for example, if you scan two assets and they look identical and they have two different IPs, it probably is a single asset. So we merge all that together at that point. Then we take the current state of the inventory for that site and say, “Okay, here's a list of the 10,000 devices currently in the site. Let's merge the new data into that.” And that process is horribly complicated and has all kinds of fun corner cases to work around weird network devices and bugs and things like that. Like, for example, we don't want to merge devices based on a MAC address that’s it's actually a failover high availability MAC or a virtual mac, because those are actually duplicated across the environment. So there's tons of tons of little corner cases that we deal with on that side.

If we find a device and it's IP change, we have to figure out what its previous IP was, merge that, update that, etc. It's this huge kind of state table merge that happens on the backend. Once we have kind of the new kind of clean version, we have a new version of the site, kind of a new state of what the network looks like. We also have a change log that we've built which shows how each asset changed. What ports are open or closed? What IPs have moved around? Things like that. And then we go synchronize that back to the database.

So the database store in this case, it's just Postgres. We use a combination of JSONB fields to do kind of columnar store when we need to and then kind of your standard RDBS normalized table structure otherwise. So kind of using a mix of those JSONB fields with your kind of standard database model has been really flexible and it's been better than trying to use something like a NoSQL or a mix of DynamoDB and RDS. So we've had pretty good luck with that so far performance-wise, query-wise, etc.

[00:28:37] JM: Are there certain kinds of networks that are particularly hard to index like maybe networks where there's like lots of edge devices?

[00:28:45] HDM: That's a good question. One of the torture tests we use Rumble is we'll take a random country and then we'll scan it with Rumble and we'll see whether it falls over. So good examples of that are we scan Iceland for kicks. It's a small country. So about four 4.3 million IPs. That was about a year ago. You find about 80,000 live devices from that set. More recently we decided to scan all of Greece, which is about five million devices, and we had a pretty good decent – I think about 800,000 unique devices come back from that as well.

So, really, challenges of scale. There are not so much weird devices in port 40 and things like that, because the engine's pretty good about figuring that stuff out today. Really the hard part is when you're doing a single site correlation of hundreds of thousands of assets trying to find things that are the same asset. Like we'll find external network routers that are actually bridging IPs and multiple /8s. Like you'll find a router that actually has IPs across like 200 different

interfaces across a huge number of networks. And you can only tell that they're actually the same device by scanning that entire IP space at the same time.

So some really cool stuff that we do on that side. You do end up with some kind of strange fingerprints in cases where a device is port forwarding enabled or things like that. For us, probably the hardest networks are ones where everything is firewalled, no ports are open and you don't have layer two visibility. So in those cases we either recommend deploying an agent inside the actual VLAN itself or providing SMBv3 credentials so that we can pull the switches and provide at least a MAC address database information as part of that.

[00:30:07] JM: So you said you could scan the entirety of Eceland and it's only 800,000 devices?

[00:30:14] HDM: Iceland's only about 80,000 responsive assets at a given time. This has about a year ago. We did a recent scan, but I don't know the numbers. Greece was about 5.3 million IPs, IPv4 addresses. And we found about 800,000 responsive IPs last time.

[00:30:27] JM: Isn't that really low?

[00:30:30] HDM: That's not too bad. That's about the normal utilization rate you see. For most IPv4 networks, you're probably looking at 20% to 30% hit rate and it depends on the network for sure. So networks are really dense and networks are really sparse. In the US, in particular, there's large corporate subnets where there's really nothing visible at all. You'll have an entire /16 with nothing in it. Just because the IPs are being used inside, behind the firewall, therefore no traffic gets routed into them. So it really depends on the network, for sure.

[00:30:57] JM: So when you scan an entire – So you're typically scanning entire IP address ranges or you start with a single IP address and you branch off from there? What's your like kind of input variable?

[00:31:12] HDM: Most folks start off with scanning the networks that are directly connected to where their agent is first. So if you download an agent, you kick off a scan, just scan your local subnet. And from there you can say, “Okay, you want to scan the rest of RFC 1918?” Like we have some customers who scan the entire private IP space every single day, and that includes the entire 10/8 all your /16, your /12 or 172. So depending on how many agents you have, it can take quite a while to do that, and you have to make sure you don't overload like your intermediary switches. But one thing we spend a lot of time doing is making sure that our per endpoint traffic is really friendly. Like we don't knock things over. It's safe to scan clinical equipment. It's safe to scan really low and embedded devices with a scanner because we spread the scan load across the entire IP space at once as evenly as possible as part of that.

[00:31:57] JM: How often should one of these scans be run? Do you usually run it every day? Like, why not?

[00:32:03] HDM: Yeah. We have folks who run scans back-to back-kind of in continuous mode. So they get alerted whenever new device shows up on the network within five or ten minutes in some cases. We've other folks who do daily scans at a certain time. One of our retail customers just kicked off a daily scan of all 220 plus stores every day at noon because they've got a nice good daily feed coming in. And those scans take about 20 minutes to run and you consolidate it back pretty quick. We have other folks who do weekly scans. Kind of anywhere in between – We definitely recommend probably daily is about the right fit for a lot of organizations.

[00:32:34] JM: I'd like to know more about the engineering behind Rumble. Can you give me more about the – Like what are the difficult problems that you're struggling with right now?

[00:32:43] HDM: Yeah, sure. So everything's written in Go. We use Github as our backend source code repositories. We use GitHub issues for tracking. My general view of development is that if the more tools you have to use to get the job done, the harder it is to keep track of everything. So, sure, when you get into larger organizations, you need to have a Kanban board. That's separate from your ticket tracker. That’s separate for your codebase. But with small

team and trying to move quickly, we try to stick everything even if it's not the best tool for every job you want to do. Having them in one place really does help.

So lots of wiki entries, lots of issues, lots of GitHub projects. Just trying to use as much of the platform that we possibly can to manage your development cycle. So the way we generally work today is we do monthly kind of minor point releases or certain monthly minor releases and sometimes daily or multiple per day point releases to the platform, and that includes building out new agents, building out new backend components that we then push then help with fingerprinting and stuff. And then all of our fingerprinting work kind of runs in parallel with that. We're really trying to be very customer-focused. So almost all of our development time is spent on things customers have explicitly asked for. Like hopefully not going into the weeds too much here, but I feel like our duty as a product vendor in this case is to, number one, listen to our customers and figure out what they need and help them solve the problem. And then there's that last little piece which is like our vision and where we think things should go. Kind of our bets on what the future of what customers may not know they need yet. And we definitely need to take that into account, but that's all secondary to making customers happy today.

Do you have any questions about the development process itself or testing or builds or all that stuff, or I guess what would you like to hear more about?

[00:34:07] JM: I'd love to know it. Yeah, because one of the things that comes to mind is the array of potential devices is quite wide. And these devices are changing all the time. I imagine they're not updating their APIs. So it's not like your scanning process goes out of date. I don't think. But there must be issues dealing with such a wide cadre of potential targets for scanning.

[00:34:32] HDM: Oh, absolutely. We do a lot of work. As part of the code in the backend that processes all of our customer jobs, it does alerting for us if something jumps out that we don't expect. So if we start seeing a panic someplace, if we start seeing unknown responses or something. Is is kind of outside of what we expect to see? That ends up going to our logging infrastructure. That ends up bubbling up something we should go fix.

As far as keeping on top of like new device types, a lot of that is driven by users too. We have the ability to improve the fingerprint by submitting it directly through the platform. And so there's a constant queue of stuff that customers are submitting. And the way our platform works today, we've got 100 plus paying customers. We have also about 3,500 free users. Just about every small business with less than 256 assets can use the platform for free. So we have a ton of folks using the platform just kind of as a free platform for inventory monitoring of their home networks of their small businesses, small banks, things like that. And as a result, all those folks are constantly giving us feedback about how well our fingerprint is working and letting us know what a device is if we don't happen to know what it is.

So between our automated monitoring, our customer-driven kind of fingerprint improvements and a lot of the data mining that we do on our side just to make sure that we're kind of staying up to what our coverage level should be. That really helps drive our focus on which protocols to research. What fingerprints to add and things like that.

We've been fortunate as well that we have quite a few partners, and a lot of those partners have really big test labs. As a really small company, like I've got a bunch of workbenches, a bunch of gear sitting on it. But obviously we can't have one of everything. So between having customers and partners who give us access to a huge variety of equipment, that's been really helpful for us keeping up to date and making sure that we can fingerprint just about anything pretty quickly.

Would you like to kind of dive into like the fingerprinting process itself or the open source components of what we do?

[00:36:08] JM: Absolutely. Yeah, let's go into fingerprinting.

[00:36:10] HDM: Okay, sure. So about half of the fingerprint that we do in Rumble is coming out of a centralized project called Recog, and that's open source. It's run by Rapid7. It's github.com/rapid7/recog. And that's basically just a set of XML databases that are full of

regular expressions. And it's kind of a first exit process. When you give a protocol, let's say, it's HTTP. You take the HTTP server header that comes out of the HTTP response. You'll start at the top of the file and you go through every fingerprint till you find one that hits and then you'll say, “Okay, great. This match. Here's the different facts that are being asserted about that device.”

So, for example, if you see Microsoft I/7.5, you know it's probably this version of Windows and it's probably manufactured after this date. There's a lot of information you can kind of infer once you see that one banner. That same process also applies to MDNS/bonjure of ID. It applies to TLS subjects, TLS issuers, FTP banners, SMTP banners, SMB native, MAC deals, native OS fields. So there's a ton of different protocols out there, all that are kind of like key valued off these pattern matching systems. Most of devices have either user agent or server header that comes back as well.

So, generally, when we're looking at a device, especially a brand new device, we want to find as many different ways to fingerprint the same way as we can. So we'll look at the web interface. Looking at just the web server, for example. You typically have a server header coming back from the response that we can build a RedX against. You've got the title of the webpage. You've got the copyright header on the bottom. You've got the favorite icon, little tiny 16 by 16 icon in some cases that's shown in the browser. And there're lots of other fields that come back like the cookies or authentication headers. It can be used for fingerprinting as well.

So just for web servers themselves, especially with TLS, you're probably looking at maybe 15 to 20 different types of fingerprints you can apply to a device just off a single service. And as you start looking to other services, things like sip, things like UPNP, TFTP, SSDP, FTP, etc., there're lots of other opportunities to figure out that device as well.

So the idea that we look at is the more different ways we can fingerprint the device, the more accurate we're going to be if we only partially fingerprint it. And if we already fingerprinted the device three or four different ways, then the next time we add a new protocol or a new type of

signature to the platform, we can quickly synthesize matches based on the existing matches of existing fingerprints. So it's kind of a – That ball has finally been rolled far enough along that we can actually – Our turnaround time for new frameworks is much faster because there's some work that's been done.

The cool thing about Recog is it's open source. It's under like a 3-Clause BSD or 2-Clause BSD license. We've got a lot of different companies contributing back to it. It's actually a project that I started working on when I actually worked at Rapid7. And that project itself started off as being the fingerprint database for the Nexpose vulnerability scanning platform.

We extract that out from the private code base, made it open source. We made it a fingerprinting backend for the Metasploit project. And then when I shifted teams to the project sonar and research team, the idea behind that team was that we'd scan the whole internet all the time every single IPv4 address with different protocols. We then took all the data we gathered from scanning the internet almost every day and then turned that into fingerprints for Recog as well.

And then since leaving Rapid7, starting Rumble, I've been using Recog as well contributing back to it from within Rumble itself. And we also have a bunch of proprietary stuff that we do as well and some heuristics that we use to pick the right fingerprint based on what we do. But at least half the work we do right now is happening in that Recog open source database.

[00:39:16] JM: And tell me a little bit more about the open source interaction with the company or what's the open source contribution path.

[00:39:23] HDM: My history is mostly working in like full open source and/or open core in a lot of cases. This is more of the opposite of that. We've been doing the core of the platform as closed source, but a lot of our peripheral libraries and peripheral fingerprints are open source. So a good example is Recog, we contribute tons back to this fingerprint library. We also maintain a MAC address fingerprint database as part of our GitHub. And we do a lot of stuff contributing back to things like SSL libraries, other kind of Go libraries that are used out there.

We've got some API tools we publish as well. So it's a little bit different from kind of the Metasploit days where it's really open core with proprietary on top. This is kind of the inverse of that. But so far so good. We're able to contribute to some shared libraries and share fingerprint projects that are used by not just us, both our company, other companies as well independent researchers.

[00:40:09] JM: Do you also try to focus on visualization of the network? Like the linkage points between different devices on the network? Or are you just focused on indexing the things in the network?

[00:40:21] HDM: Oh, yeah. Visualization is a huge part of it. I feel like we try to provide people a unique view of the network they may have not seen before. So if you think your network consists of a couple routers, a couple switches, a bunch of Windows devices, and then you look at the Rumble output and you're like, “What the heck is that?” Oftentimes we're showing you layer two topologies you realize existed. We're showing you layer three bridging because of multi-home interfaces connecting disparate networks. We're showing you kind of the grid report of the entire summit utilization showing you like how your IP space is actually used by what types of devices. There's a lot of really cool visualization we can do. That's one of the challenges that I struggle with, because I'm a really terrible UX developer. So that's the thing I'm probably the worst at right now. It's how do I take all the ideas I have for visualizing this stuff and actually get them out on the screen. And that's hopefully someone we can hire for and do better at going forward.

But some of the cool things we can do today are, as part of the layer 2 topology analysis that we do through SNMP, we can show you like how each switch is linked to each other switch. How each individual asset is tied to each switch port? So we can tell you like what port each device is hanging off of. Because Rumble's really good at identifying multi-home devices both by directly discovering secondary interfaces on a device and by inferring that a device is multi-home by uniquely fingerprinting and matching it to a device in a different subnet, we can show you a layer 3 graph that looks absolutely nothing like traceroute.

We'll show you, like for example, if you've got a corporate laptop that is on the wireless network that you then plug into the wired port, we can show you that that laptop is now acting as a bridge between those two networks. So we can identify a lot of segmentation problems that are relevant for folks who are in the PIC space where they really have to keep their car data holder environment separate from everything else. Or other organizations whether you want to keep like their OT network separate from the IoT network and vice versa.

So some of the cool layer three topology analysis stuff we do is really useful for finding those segmentation problems. And when you start getting kind of the higher level looking like our subnet grid reports, you start to see patterns pop up right away. Like, for example, if you see – You're looking at a /16 at a time and you see a big stripe of voice over IP devices, you're like, "Hey, those really shouldn't be there. That should be part of a VLAN for voice instead." It makes it really easy to identify configuration problems or architectural problems that are really difficult to find just by looking at an index list.

[00:42:25] JM: Is there any particular reason why somebody hadn't built Rumble before you started it?

[00:42:30] HDM: Probably because it's hard. If you look at kind of the history of people doing active scanning base discovery, majority of the folks who started off in the space doing authenticated scanning moved to authenticated scanning. They decided that our customers really want to have software inventories. They want a deeper view of what's out there. And instead of becoming really good at identifying things that they couldn't log into, they became really good at pulling data out of things they could log into. And as a result, most of the tools that do authenticated scanning today can tell you a whole lot about assets they can authenticate to and almost nothing about them otherwise. They're very binary in that sense.

And as the kind of internet's changed, companies have become more security aware. Internal firewalls are more often, more consumer IoT devices on corporate networks. The amount of devices you can authenticate to is actually shrinking. Oftentimes firewalls prevent a scanner on the local network from logging into every Windows device, or the same credentials don't work

in every device because you have a lot of employee laptops or you've got multiple active directory domains and they don't all jive and they don't trust each other.

So because of how kind of complicated IoT networks have become, authenticated active scanning has become more difficult over time. And as a result many organizations have moved towards API-based discovery, pulling central databases, pulling DNS logs, doing passive analysis, all that other stuff. So we kind of went the other way on that. We decided like, “Hey, this is a hard problem, but this is a fun problem. This is something we could actually go solve.” So taking the perspective of kind of the security pen test red team mindset and saying, “Hey, we really want to figure out what these devices are,” and do the same level of effort that would take to write a really reliable exploit or write a really good vulnerability signature and instead taking that effort and applying it towards device identification. I think that's why we're doing well. I think it's why we're able to do a lot of work. But it's also why there's very few firms who do it that way. It's a hard job to do and it's something that I feel like we're kind of uniquely qualified for because of our backgrounds. And it's really difficult to get into if you don't already have a very low level networking background to start with.

[00:44:20] JM: Well, as we wind to a close, do you have any predictions for the near future of Rumble or security in general?

[00:44:29] HDM: It's going to be harder before it gets easier, I think. I mean you look at some of the stuff that's happening these days with supply chain attacks and so on, and it's a tricky world out there. And in the past I think a lot of folks could count on their infrastructure being consistent. So if you're a DevOps shop and you did things a certain way, you can kind of plan for everything to be done that way. And if you’re a Microsoft shop and you had an active directory domain and you kind of manage a certain way, that's just how things are done. And going forward, I think we're going to have a whole lot more hybridization. We’re going to have a lot of mixed cloud VPN links, remote networks, BOID, home users, especially with COVID. So we're just going to see networks become more and more chaotic over time, and that means mixes of assets you can authenticate to, assets you can't manage, unmanaged, OT, vendor managed third party. It's chaos. But it's good. It's also the evolution of IT networking and

technology in general. So I guess that's my only real prediction, is that things will get much more chaotic, and as a result a lot of the tools will have to adapt.

[00:45:24] JM: Awesome. Well, thank you so much for coming on the show. It's been a real pleasure talking to you.

[00:45:28] HDM: Oh, thank you very much, Jeffrey. I appreciate your time.

[END]