Title: Managing Security Vulnerabilities Based on What Matters Most
Transcript

Part 1: The Challenges in Defining a Security Vulnerability

Julia Allen: Welcome to CERT's Podcast Series: Security for Business Leaders. The CERT Program is part of the Software Engineering Institute, a federally funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. You can find out more about us at cert.org.

Show notes for today's conversation are available at the podcast website.

My name is Julia Allen. I'm a senior researcher at CERT working on security governance and executive outreach. Today I'm pleased to introduce Art Manion, who leads CERT's Vulnerability Analysis Team. Art and I will be discussing how security vulnerabilities have evolved and how business leaders can more effectively manage the thousands of new software vulnerabilities that are reported each year. So welcome Art, glad to have you here with us today.

Art Manion: Great, thanks, happy to be here.

Julia Allen: So let's start out with some basic terminology. What is a software vulnerability?

Art Manion: Okay, so there are lots of different sort of definitions of vulnerability, and when we, when we're speaking, when the CERT Vulnerability Analysis Team is talking about a vulnerability, it's a software vulnerability. It's most often some sort of coding defect or a bug that allows, for instance, an attacker to break into a system or violate some kind of security policy.

Things we look at are do the conditions allow there to be some sort of impact? Does the attacker gain something? Is there a consequence, something that's not normally allowed? We talk about explicit security policies. There may be a rule on a firewall that says "This port or service is not allowed to cross this boundary."

A couple of other interesting elements to vulnerability as we describe it: a vulnerability has to be caused by a human; an engineer or a developer who wrote software, created a bug, or perhaps a designer or a software producer configured something in a certain way, or failed to configure something in a certain way, and that resulted in a vulnerability.

And maybe the most, one of the more interesting pieces, is this idea of sort of a changing environment. An example might be a protocol that was designed many years ago. It was designed maybe when the Internet was not so large or such a hostile place, and the protocol was designed to speak among trusted peers and trusted nodes. In today's Internet, that protocol – if it has no notion of a threat against it, no authentication or no encryption – that protocol probably won't stand up very well in today's Internet. Nonetheless, we see old protocols and old designs that are now trying to survive in today's Internet, and the changing environment around the software has created vulnerabilities where there sort of weren't any beforehand.

So we have sometimes roundtable discussions about "Is this report a vulnerability or not? How should we respond?" But those are some of the things we think about when we try to define a vulnerability.

Julia Allen: You know, that's pretty interesting, the last category that you described. Because sometimes we might have a piece of software that's as vulnerability-and defect-free as we can

Copyright 2008 by Carnegie Mellon University

make it, but yet when you put it into an environment and it's interfacing with other software and other systems, just by virtue of that new environment or that new context, it can be vulnerable, right?

Art Manion: Absolutely. A clear example that I've come to understand somewhat recently is probably control system protocols. These are things like DNP3 or Modbus – protocols that were designed maybe to run on serial lines, directly connected between the terminal and a control system of some type – a pump or a valve or a machine. And they were designed to run within a factory, within a boundary, with a point-to-point serial interface.

For obvious reasons – cost savings and remote access – these protocols are being ported, or have been ported, to run on top of IP or TCP/IP. So here you have this protocol that's designed for deterministic behavior, for speed, for reliability, and now it's on top of TCP/IP, and this protocol maybe has no authentication, or a very small, simple packet can really control the device it's being sent to, change memory, rewrite the configuration. And that's a very, I think one of the most clear examples of taking a protocol that was designed with one set of environmental conditions, changing the environment it works in, and all of a sudden now the protocol is not as secure as it should be anymore.

Part 2: The Shifting Vulnerability Landscape

Julia Allen: Well you know that's a nice segue into what I wanted to ask you next which is could you say a little bit about how the characteristics of vulnerabilities have changed since CERT started tracking them and, in particular, why this should be of concern to business leaders?

Art Manion: Probably one of the larger trends we've seen in the past three, four years in vulnerabilities has been related to really just changes in software and the prevalence of web-based software – web applications, software as a service, Web 2.0 types of applications. We're seeing a lot more vulnerability reports that are specific to web applications – cross-site scripting, SQL injection attacks, cross-site request forgery. I'm actually behind on some of the new names and acronyms for these types of attacks.

But there's definitely been an upswing in those types of reports. Now I believe some of the reasons for the upswing are there is a lot more web-based software deployed. It's fairly to easy to make these sorts of mistakes and cause these vulnerabilities; fairly easy to find them too, if you're willing to scan websites for these types of vulnerabilities. So that's been a trend in the types of vulnerabilities found. There's some evidence that says that sort of the classic buffer overflow vulnerability just sort of stayed at a static rate, over time, but that these web-based vulnerabilities have sort of gone up.

Perhaps something that's more interesting is the types of vulnerabilities that attackers are exploiting. So if you just look at vulnerabilities, maybe there are more web app vulnerabilities, but if you look at what vulnerabilities are being selected by attackers, that has shifted some. When I started working at CERT, and this is maybe seven years ago, it was still a time when a network service, a UNIX or even a Windows-based network service, might have a vulnerability, and worms and attackers would target servers that were listening on a port and try to break in, and the worms, of course, would try to break in quickly to many machines and multiply.

Today we're seeing sort of a different vulnerability selection by attackers, and I believe this is mostly related to the attacker trend, sort of a threat trend. I believe attackers still try to compromise servers and get footholds in places, and you do have to consider what type of adversary you're facing. But a very broad section of attackers now are after personal information – credit card

Copyright 2008 by Carnegie Mellon University

numbers, bank account numbers, passwords, access to other websites. And in order to get that information they're attacking personal computers, and there's usually some phishing component.

So what's prevalent today are attacks against vulnerabilities in web browsers. It's very common to see a web browser vulnerability show up in real live attacks. Plugins for web browsers, media players are often targets of attack. And the reason is that if an attacker can convince someone to visit a website, they can very quickly, if that's an unpatched browser, attack that browser or attack a media plugin, and then very quickly, by soliciting someone to come into a website, click on a link in some spam or some phish URL, they've attacked the machine, they've installed their keylogger, their screen capture, their data exfiltation system. And that's a very distinct trend we're seeing among attackers these days. So instead of the firing away at these servers, listening on a certain port, it's send out mail to convince users to come visit my malicious website; then attacks their browser, attacks their plugins. Perhaps I send a document that's in a format that has a vulnerability and convince you to open a document. Again, I've convinced someone to run my malicious code and I can do what I want to their computer.

Julia Allen: So then – I mean you've mentioned a few of these – but what do you see? How does this show up as impact to the business?

Art Manion: You're still ending up with compromised machines. Whereas perhaps it used to be more likely that a server would be compromised, maybe today it's the average workstation that's compromised. Now I say average workstation – that could mean a lot of different things to a different business. To a home user it might mean their family photos are at risk. It might mean some bank account information is at risk, which is of course a major concern. To a business this is going to vary to some extent.

Basically the assumption that needs to be made is if an attacker has installed software on the computer by tricking a user to visit a website, that attacker can do anything that that user can do. So if that operator of the workstation has access to a sensitive internal financial application or internal database or can send email as the business, the attacker could do those things.

Julia Allen: In other words the attacker has all the rights and privileges and accesses that the user would have by virtue of having compromised that user's machine.

Art Manion: Yes, absolutely. There still comes a question as to does the attacker realize what system they're on? Were they targeting that business? Did they have a certain goal in mind, a certain internal asset maybe? These widespread attacks are still very automated and generally the attacker is sort of casting a wide net. And they may not know individually that this one out of 10,000 machines that I've compromised happens to be at this particular business and I can gain this particular asset at this business. So in that case, even though the machine's compromised, it might just be used as part of a botnet to send spam or further attacks.

If the attacker is targeting a certain business it's a much different story. Then they might be after something specific – have some direct knowledge of what they can gain by attacking that particular workstation at that business. So targeted attack is a different problem than the widespread attack.

Julia Allen: Well, and as you said earlier, chances are with a targeted attack the attacker does have much more intent and knowledge about where they are, what machines they've compromised, and what their ultimate objective is.

Art Manion: Yes, absolutely.

Copyright 2008 by Carnegie Mellon University

Part 3: Determining which Vulnerabilities to Pay Attention To

Julia Allen: So how are vulnerabilities typically discovered and communicated, do you find?

Art Manion: Well so we do something we call public monitoring, or open-source monitoring. And we see that most vulnerabilities – we're counting around 7(000) or 8000 new reports per year, in the public light – most of these are reported on mailing lists or blogs or websites. There are a number of other organizations that also sort of monitor and collect this information. So there's a fairly well developed community that watches for these reports and logs them and writes brief entries, catalogues them. We see reports that come directly to us. And researchers who find things also sometimes will go straight to a vendor and report a problem directly to the vendor of the affected software. And that's – reporting it quietly to the vendor or quietly through a coordinator like CERT – we usually term that *responsible disclosure*, which we advocate; giving the producer of the software a chance to fix the problem and protect their customers before disclosing the whole thing to the public.

There are certainly researchers whose job or hobby it is to go find vulnerabilities. There are people who find things by accident. I used to be a system administrator and maybe if I was troubleshooting some behavior I might come across a vulnerability; we've seen that happen. There are lots of tools these days – fuzz testers, source code analysis tools – that can find problems. So there are lots of ways to find them. There are a variety of ways, of reasons, people look for them. Publishing is still often a mailing list or a website.

Maybe one of more interesting trends or changes in the last several years has been that a so-called zero-day vulnerability – a vulnerability that is only known by a small community, perhaps the researcher who found it and maybe a few others – that has monetary value to the point that there are security firms, and arguably a black or a gray market even, that will pay for this vulnerability information. So now there's an actual incentive. If I find a vulnerability I could be paid to find it and report it to someone, which is something that didn't obviously exist when I started doing this work.

Julia Allen: Well that's pretty interesting. Some of the normal market incentives might apply in this case then?

Art Manion: Absolutely. In fact, it's a very interesting study in economics. I've read a couple of papers that try to figure out how this market works. There are supply and demand issues. If there are lots of vulnerabilities in a certain type, supposedly the value goes down. It's very interesting too, when the vulnerability becomes widely known, its value drops significantly. So a premium price is paid for a vulnerability that is only known to the researcher and hopefully the single person that they are selling to, or the buyer. If the researcher sells it to many different buyers, and the buyers know about that, the value drops. I'm not an economist, so I shouldn't go too far here, but there's some really fascinating economics factors that go into play here.

Julia Allen: Very interesting. Yes, the old supply and demand equations come to mind.

Art Manion: Yes, yes.

Julia Allen: So obviously addressing all software vulnerabilities, kind of in the same way as being 100% secure, is not possible and even if we could this likely isn't a good use of our business resources, right? So how do I figure out – if I'm a business leader or if I'm a manager of IT, or if I'm doing a major software development project – how do I figure out which vulnerabilities I should pay closest attention to?

Copyright 2008 by Carnegie Mellon University

Art Manion: This is an interesting problem and it's one that my team faces every day. I mentioned earlier we're collecting 7000, 8000 a year. We don't have the resources to track down every one, figure it out, analyze it carefully, make sure we find all the vendors of that software. So whether you are someone on my team, a system administrator, a patch administrator, somebody responsible at your business for making sure your systems are updated and secure, you're going to face a similar problem.

Our current thinking – or our most advanced, I'll say, to this date thinking – is that at the scale of 7(000) to 8000 per year, there has to be – you need some computer help. You can't just throw bodies and FTE (full time equivalent) hours at the problem. There needs to be something systematic, a computer system involved. And there's a component that's really important, which is how your individual business or your individual network values certain assets and operates certain parts of the network and certain parts of systems.

So, for example, if there's a vulnerability in a particular database, a piece of database software, one business might run all of their business critical applications on that database platform. That's clearly going to probably be a high value asset and maybe a higher risk, higher severity vulnerability to them. If very simply a second business doesn't run that database software, then any time they spend analyzing that vulnerability, past the first couple of minutes is really poorly invested time, security-wise.

So I guess the approach we're looking at is sort of an expert system, and we're using a little bit of a decision support, a very lightweight decision support model. So that if vulnerability information is published in a computer readable format – so important characteristics of the vulnerability can be understood by a computer system – and then the person making the decision on how best to respond needs to have some knowledge of their network, their assets, what software they use, at least a rough idea of the value of those assets, how much risk is involved in having one compromised or damaged, or if it's service, if it's not available how bad is that? And a system that combines these things and then can basically tell the operator, "Out of 20 new vulnerabilities released yesterday, three or four of these you really have to look at today. Assign your people, assign your resources to look at these three or four because they really matter to your business." Maybe there's 11 or 12 left that if you have time you should investigate them but they're not that Important. And the remainder, don't even look at the report. You don't run this software, or if you do, the way you run it, there's just no possible impact. It's not worth your time to spend your energy on those particular vulnerabilities.

Julia Allen: This is kind of a nice consistent message in terms of some of the other research initiatives we've talked about in the podcast series – our governance and risk management and operational resilience work where you really need to get a good handle on what's most important, what the threats are, how those threats might be realized in terms of taking advantage of vulnerabilities. I think the nice twist or value add that you've put on this is then how do you kind of, in some ways, automate that in some kind of a decision support way so that it makes it a little bit easier for someone in the role of picking and choosing which vulnerabilities they pay attention to, they can do that in a much better informed way.

Art Manion: Yes, right. Our system, or our research so far, is really driven by our experience, which is it is a management problem, it's a vulnerability information management issue. Going and patching the system, that's technical. Making a firewall change to protect your systems, also technical. But managing this information, figuring out, out of this big pile of data, which bits matter to my business, that's really the first focus. From there you can choose how to implement your technical response. But first things – the first thing to do is identify what matters the most and focus your efforts on that.

Copyright 2008 by Carnegie Mellon University

Julia Allen: That's just good business sense.

Art Manion: Yes.

Julia Allen: So are there, are you finding, are there practices or solutions that could help organizations get in front of this vulnerability issue, which tends to be more of a reactive or a response issue? Something that might help them reduce the number that show up in their production systems in the first place?

Art Manion: Lingering slighting in the response side there are commercial alerting services, there are ways to be — to pay for a fee, basically a subscription that will tell you, in a reactive sense, what's coming out every day, every day, every day. There are filtering mechanisms. So there's some systems in place now to sort of help with the awareness and the alerting part of it. But, as you're getting to here, that's a never-ending stream of information and vulnerabilities, and what we'd really like to see is software that doesn't appear in one of these feeds, software that doesn't have vulnerabilities or it doesn't have as many vulnerabilities in the first place.

So looking at it from that angle, looking at it from the "don't create a vulnerability in the first place" approach, there is a secure coding initiative at CERT. Part of that initiative involves a secure coding standard. Any technical help, software development processes, code analysis tools that can identify defects that could lead to vulnerabilities, things that are going to help businesses and software producers not create defects that are also vulnerabilities in the first place is a more proactive approach.

I think, from my team's point of view, we try to relate what comes out as a vulnerability, in a response fashion, we try to tie that back to a secure coding rule, if we can identify one, to really make it clear that "A programmer should not write this line of code in this way, and if you do that, here's something that could happen later. There's a vulnerability report, a vulnerability document, lots of vendors affected, patches issued."

All the guidance I've seen and all the numbers I've seen are that it's very costly to fix a defect once it's out, it's deployed, it's on customers' systems. The closer you get to the creation of that code in the first place, the closer you get to fixing that vulnerability or not having that defect, the cheaper it is for the software producer.

Julia Allen: You know we always have such a difficult time doing return on investment or other types of cost-benefit analysis, but with some of what you've just described, I could foresee a time when you could say, "Okay, this is how a vulnerability is showing up in production and operation today." If you track that back to making this particular change during software development, you can get a direct correlation of how much it costs to do it during code versus what it costs you to address it as an operational issue.

Art Manion: Yes, I'd love to spend some time trying to work on those numbers. It's not my day job, unfortunately, but that's — we're trying to provide data that would help that sort of analysis.

Julia Allen: Well Art this has just been excellent. I think a real good introduction and description and kind of a very tangible and practical set of things for our listeners to think about. I know that you're doing this as a very — there's some very active research elements to the vulnerability analysis agenda. Do you have some sources that you'd like to point our listeners to where they can learn more?

Copyright 2008 by Carnegie Mellon University

Art Manion: Well you mentioned, of course, the CERT website, and from there a couple of interesting places are – we've got a blog recently, so there's a Vulnerability Analysis Team blog, which is nice because we're able to put out less structured data. Usually we have to publish a vulnerability document, very structured. Now we can talk with the blog a little more broadly about trends or opinions, things that we're seeing going on.

The Secure Coding Initiative is based on the CERT website. Most of the work for my team is published as a US-CERT document. I'll pitch our system, our vulnerability management support system. The acronym is VRDA, v-r-d-a, Vulnerability Response and Decision Assistance, and there is a paper also on the CERT website that outlines how that system works. And hopefully more to come in that research field.

Julia Allen: Well that's great. We'll include links to all of those in the show notes. And I'm so very appreciative of your time and your expertise, sharing some very important information that I hope will bring listeners to your site and stimulate some more conversation.

Art Manion: Great. Well thanks for the opportunity.

Copyright 2008 by Carnegie Mellon University