**RedZone Podcast Episode #76**

Speaker 1: Welcome you to the show today.

Jeff: Thanks. Great to be here.

Speaker 1:

[00:00:30] I'm very excited to have this conversation, because I definitely want to talk about your company and your product. Recently I ran into you at the RSA conference, and listened to you in some of the awards that you and your company have generated with your product, but I thought maybe one of the ways we could start is just, let's talk about your beginnings within security. Let's talk about entrepreneurship. Let's talk about the story around this, because I think that's as meaningful as where you are today with your products and your solutions you're bringing to market.

Jeff:

[00:01:00]

[00:01:30] Sure. That's great. I got started in security back in the mid '80s. I was in high school. If you remember back then, there was copy protection on a lot of software. I wanted to back up my games and whatnot, so I found cracking copy protection to be a lot of fun. It's like puzzles. You work through the different layers of protection, and eventually that code has to run, so you can figure out a way to get a copy of it out of memory. I did that for a while. Got good at it. That forced me to learn a lot. Learn about disassembly, learn about operating systems, learn about really, the beginnings of learning about instrumentation, which ultimately became my whole career, but we'll talk about that in a little bit.

Speaker 1: I noticed that. You went to George Mason and I guess that was undergrad and UVA, or maybe it was UVA undergrad.

Jeff: Yeah.

Speaker 1: George Mason.

Jeff: I got a masters at George Mason and then I went to Georgetown Law.

Speaker 1:
[00:02:00] Which is interesting, because I never heard ... Human factors. What is human factors? I noticed your first job incorporated human factors. Was there a tie in to the intellectual property piece and kind of reverse engineering protection around the games and such?

Jeff:

[00:02:30]

[00:03:00] Not so much a connection. I didn't have a great plan, actually. At UVA I studied computer science and psychology. I was trying to put together an interdisciplinary major in artificial intelligence, but I quickly figured out that I was about 30 years too early. I went off, I started ... I got a job on a big navy project, and it had a huge user interface, that I took over. 600 screens or something. I worked on that for a while. That accidentally got me into security as well, because one of my jobs was to port that user interface to a new, trusted operating system, called Compartmented Mode Workstation. Eventually it became ... The one that won was Trusted Solaris. I got real into security there.

[00:03:30] But I decided to go back and get a masters in human factors engineering, which today you'd probably call it like UI, UX design kinds of considerations. I was into computer/human interaction then. But what's interesting is both security and user interface, you really have to work with a broad spectrum of folks on a technical project. It's not just go off and do your piece. You've got to work with everybody, and I think that's an interesting similarity between those disciplines.

Speaker 1:

[00:04:00] Yeah. Gosh. That's ... We're, as you may know, I'm heavily involved in the security side, and what I find the most challenging is the application side. I just, just because of the different teams and then securing code. The quick story around that is, when the early application firewalls came out, it was, this is so cool, we can secure the ... We can just put a front end on the code, and just have a device sit in front of a suite of applications. Oh my gosh. Our phone has never rung so quickly, because it ended up having to be ... You'd have to spend as much time with the developers fixing the code problems than in tuning the front end firewall.

[00:04:30]
Jeff:
Yeah, that's right. I always say that network security is like Legos. Everybody's using the same blocks, basically, and so a lot of the problems are similar, but with application security, it's more like clay. Everybody's building this custom thing, so if you try to just put a generic security device in front of it, you're going to have to spend a ton of time tuning it and reconfiguring it, and anytime they change the clay, then they've got to ... You've got to reconfigure the, what about the firewall? It's really messy.

[00:05:00] The problem is, it's too far away from what you're trying to protect.

Speaker 1:

[00:05:30] Interesting. I definitely want to get into that side, because I know, it'd be interesting, I really want to hear about what you're currently doing with your business, but I want to hear a little bit about ... In the early 2000s, you started, because this all connects together, so it's funny how your career, even though it's called human factors, is really UI, UX, little bit of AI, maybe it would be called today, and then you went into ... Then you cofounded [OWASP 00:05:31]. Maybe you can talk about what that is, as a general thesis, what you're trying to accomplish there, and maybe we can start off there.

Jeff:

[00:06:00] If you rewind back to early 2000s, there was this mailing list that got started, and it was basically a bunch of folks that were interested in application security. It was just really kind of new, so we were trying to pull together some best practices. We were trying to pull together a guide for testing and building secure applications, and just collect some of the knowledge that's out there.

[00:06:30]

[00:07:00] At that point I'd been doing application security work for several years. I started an application security team at Exodus Communications back in the late '90s. We'd built up some best practices, and tools and techniques for doing some things, but it was still really, really early. On the list, some folks started talking about building an application to allow folks to experiment with vulnerabilities. Sort of an online learning environment. They were going to build us one from scratch, and I had already built one for secure coding courses that I'd been teaching. I said, "Hey, you know what? If they build one, everyone's going to use that. I want them to use mine. Why don't I just make that open source?" I contributed it, and people really seemed to like it. That's a project that's called Web Goat. People are still using it to learn about application security. There's a bunch of other ones now, but one of the first.

[00:07:30] It was really, it was a great feeling. People were raving about this tool and having fun with it, and I know a lot of people that got their start in application security from using Web Goat. I got a little addicted to doing open source projects. It's incredibly rewarding.

[00:08:00]

[00:08:30] Later that year, I had an idea for a sort of a top ten document, and I wrote up the first OWASP top ten, along with Dave Wickers, and we contributed that to OWASP, and that went crazy. We got slash dotted, and OWASP went down for several days. It was growing pains early, but it was cool, because now, at that point it was hundreds of thousands of people were using our work, and later it's millions, and so on. It was great. We started doing conferences at OWASP, and that was, built our community even more, and at some point the guy who had been running it, named Mark [Curfie 00:08:18], he decided it wasn't moving fast enough for him. He said, "Hey Jeff, do you want to be chair of this?" Because I'd been contributing things. I said, "Well, I don't know. I've got a job, but okay, I'll take it on.

I became chair, and I decided to try to make OWASP a self-sustaining organization, so I made it a 501C3, put a board in place, set up chapters, programs, and put a lot of the governance stuff in place. I wanted to sort of transition it faster, but eight years later, I was able to step away and say, "This is a going concern. I don't have any concerns that OWASP is going to stick around," and it has.

[00:09:00]
Speaker 1:
That's great. Basically you put some legs around it so that it could be self-sustaining, and is the little engine that could, so it will be able to self-maintain itself with its own revenue stream, so that it can keep adding value back to the community.

Jeff:

[00:09:30] Yeah. You asked about entrepreneurship, and really I think that's the core of entrepreneurship, is building something, an organization that will last. OWASP is a very different kind of organization, because everything is free and open, and the revenue is sort of secondary. We do get a little bit of revenue from conferences and selling shirts and things like that, but really, almost all the labor, all the effort, all the good work that comes out of OWASP is done by volunteers. It's a different kind of organization, that really appeals to me.

[00:10:00]
Speaker 1:

[00:10:30]
The interesting piece about that is the development of communities over the past several years, but certainly at the genesis in the 2000s, is just these communities of support around Linux and [Ilwasp 00:10:16] and the list goes on and on, but even for profit companies are needing to really look at eliciting the support of communities that have, like Kickstarter and all these different programs. I don't know if you've seen much success with for profit companies really leveraging communities, and how they can potentially do that to achieve some really success with tough projects or goals that they may have. Have you seen that much?

Jeff:

[00:11:00]

[00:11:30] I have. I think it's pretty rare, but when companies do it, I think it's fantastic. That said, OWASP isn't all a success. I think if you look at the history of application security, we've made some progress, like there's better tools. Now we know a little bit more about what it takes to find vulnerabilities and fix them, and operate applications securely, but frankly, software over the last 15 years has been exploding. It's going to fast. OWASP is kind of plugging along, but the net of it is that we're falling farther and farther behind. I really think OWASP needs to make some major changes in order to try to really become part of software development and not just this extra side thing that people do because they have to.

Speaker 1: Why is it ... With software exploding, why is ... What does falling behind mean to you, and what does it mean to fall behind with the current state of application security?

[00:12:00]
Jeff:

[00:12:30]
I knew we were in trouble the first time I saw a single page application. It was probably about 2007 or so, and I got this app that I was testing, and I was using a proxy, like web scarab, and intercepting requests, and tampering with them, and the single page app was sending XHR request, JavaScript request, in the background, one every second. My intercept window was just popping up and piling up intercepts way faster than I could deal with them. I thought, wow, this is tough. This means I'm going to have to take my testing and automate it. I'm going to have to turn it into code so that I can apply these changes dynamically as the requests are going back and forth, which is a whole different kind of paradigm than what we were using back then.

[00:13:00] Since then, there's been so many developments that have made application security harder. Then it was the explosion of libraries and frameworks. These days, developers can just indicate, "Hey, I want this library," and the dependency resolution tools, like Maven and those build tools, will pull in all the dependencies for that. Where we used to see ten or fifteen libraries in an application, now we'll see 200, 300, in a big application. The amount of code just has exploded. That makes it more difficult.

[00:13:30]

[00:14:00] There's also the explosion of the use of API's. It goes sort of along with cloud and containers, but API's are a different animal. They need different tools, different techniques, in order to test them. Now, most modern web applications are really JavaScript in the browser that connects back to a set of web services, or rest API's, on the application side. That's an area where we're way behind already. I don't particularly see that getting better very quickly.

Speaker 1: That's interesting.

Jeff: There's a lot going on here.

Speaker 1: It's interesting, because I don't know, have you read the book Exponential Organizations, by Celine?

Jeff: No.

Speaker 1:

[00:14:30] One of the big pieces he talks about is how the CIO of the future, and currently CTO's and CIO's are going to have to ... This is more from an innovation perspective, are going to have to get really strong in understanding API interaction with their data, so that they can add value to the business, and add revenue streams, and be able to pull public and private data using API's, so to your point, there is this kind of offense push in proliferation of software, but then how does our security scale at the pace of innovation ... How do you govern at scale and keep pace at the same time? What are your thoughts on that?

[00:15:00]
Jeff:

[00:15:30]

[00:16:00]
One of the biggest changes we've seen in the last five to seven years or so is the rapid growth and adoption of Agile and [Devops 00:15:09]. It's really straining application security way past the breaking point. It was one thing if you were releasing an application once a year, we could come in and do a code review and a pen test and get some results, and hopefully fix those. That all sort of worked. Now if you're releasing in two week or one week sprints, or you're doing deployments multiple times a day, there's really not time to do the kind of deep security analysis that is really required, and I'm a big fan of manual code review, and I spent decades doing that kind of work, and unfortunately it's just kind of incompatible with the way people actually build software today. That pace issue is critical. It's straining every bit of what we know about application security. I've been pushing very hard to try to get people to do application security a different way, a continuous way, as opposed to trying to do periodic scans and tests.

Speaker 1:

[00:16:30]

[00:17:00] That's really interesting. Man. I was just in ... How ... Also, there's another issue that is interesting, because I was having this conversation with a CIO, who they're building an app, and so we were in with their app team, talking about application security, then but that ... But it's not just the security of the app on the phone. It's now the app on the back end as well, and the web server, and the back end data, and how do you approach that? If you're not going to do manual code reviews, what's another philosophy that you think would work in an environment where you're building apps, but you've still got to support the back end infrastructure as well, across multiple partners, and hosting facilities? Where do you start?

Jeff:

[00:17:30] The first important thing is to, I think, get a real understanding of what matters, for your organization, for your application, and probably the best way to do that is through threat modeling, which is essentially a process of just understanding who the attackers are, what they might do that would harm the company, where the assets, or what are the capabilities that you want to protect, and then figuring out what defenses really matter.

In terms of verifying the security of that application, you should look at the threat model and say, "Okay, what are our critical defenses? Let's make sure that those are actually in place, that they're actually working, that they're correct," meaning that they're not bypassable and they don't have flaws in them and things like that, "and that they're used in all the right places.

[00:18:00] For many organizations, that probably means they need to start really with the things that are going to take them out of business. Things like injection kinds of problems, access control kinds of problems, authentication control kinds of problems. These are the things that lead to major breaches.

[00:18:30]

[00:19:00] Since you mentioned the mobile versus server side application, I think it's really important to know that most of the risk to enterprises is on the server side of applications. That's where all the data is. That's where all the customer accounts are. That's where the important resources and capabilities are, and if you look at something like the Verizon data breach investigations report, they make it really clear, there's never been a big breach that started because of a vulnerability in a mobile phone. I'm not saying that those apps don't have vulnerabilities. They certainly do. We found a ton of them. But when you really work out the impact, it's difficult to imagine somebody hacking into enough phones to really make a difference.

Prioritize your business by risk. Then work off the risks in order.

Speaker 1:

[00:19:30]

[00:20:00] I think, as we were sitting in this meeting, I was listening to my CTO, and I talked about it with him afterwards, and he basically said the same thing, which is don't pay as much attention to the end point. It's sort of your return on energy as a hacker just wouldn't be as strong, but then what's interesting from your point of view, because when you're pulling in potentially third parties into that app that would distribute data to that phone, and you're maybe pulling in HIPAA data or PCI related data into that back end infrastructure, all of the sudden you've got potentially people contributing to the risk on the back end. You're saying you've just got to do a traditional threat modeling on the back end to see where your risk is.

Jeff:

[00:20:30] The alternative is really not very appealing, right? The alternative is to not know what's most important, and to just start testing a whole bunch of crap, right? Testing little things, and so you end up wasting time testing things that don't really matter and leaving things that really do matter untested, so access control is a really good example. It's hard to test access control. It's not something you can do generically because everybody writes their own. It's that clay problem again.

We see tons and tons of apps where they've gone through some static analysis tool, some dynamic analysis tool that was written back in the early 2000s, and they think they're secure, because they went through the tool, and they fixed anything the tool found, but the tool wasn't looking for that stuff. It's really important to know what your tools are good at and what they do, and where they just have gaps.

[00:21:00]
Speaker 1:
What does it mean, access control to you, in the context you were just talking about? Just from a scenario perspective. Can you just go through that?

Jeff:

[00:21:30] Sure. Access control is pretty complicated. Authentication is easy, right? Authentication asks the question, who are you? Do I know who I'm dealing with? You ask that question once. Once you know who they are, then you know their identity, but access control is a little different. The first question you might ask is, do this person have access to the URL that they're trying to go to? There's a very high level kind of access control.

[00:22:00] Later, let's say they have access to the reporting function. Well, maybe they're trying to generate an admin report. Do they have access to that, or do they just have access to the normal user report? Even more deeply, you can imagine on the data level, does this person really have access to this data that they're trying extract, or not? There's all these different levels of access control, and it kind of mirrors the real world. You can imagine when you go go to some corporate facility, you get sort of an access control check at the guard gate when you drive in. You get one at the front door. Maybe somebody has to escort you. Maybe there's a locked file cabinet. Those are all access control checks. The problem for application security is, it's really hard to know where those are supposed to be. If you knew where they were all supposed to be, you could check them pretty easily, but just understanding the policy is actually really tricky.

[00:22:30]
Speaker 1:
This is ... It's actually, that's ... This is your traditional identity management role. The [RBAC 00:22:33], right? The rules based access control?

Jeff: Mm-hmm (affirmative)-

Speaker 1: That's really what you're getting to, right?

Jeff:

[00:23:00] that's one way of implementing access control. There's others. There's a bunch of different ways. You'd be surprised. Access control is a very old problem. If you go back, even before the orange book, multiple different access control models, from the operating system era, that have been studied. They've got strengths and weaknesses. We actually forgot a lot of that when we started building web applications. All the knowledge that we'd learned building really highly secure operating system access control mechanisms just got forgotten, and we just started implementing crazy stuff like HD access and sort of poorly thought out access control schemes for web applications. That's still the case. Java EE access control is still kind of a mess.

[00:23:30]
Speaker 1:

[00:24:00]
It's interesting, as we talk about this, and it's in many respects probably the holy grail of the future of a lot of security pieces, because like in a financial system, obviously a CFO would have certain rights to access certain pieces of the application than a new clerk that was just doing data entry in the accounting system, so there's different role, that person would have different permissions that they're allowed, but that's a financial accounting system, and that's as much of a business issue as it is someone setting that system up with the right permissions maybe technically in the back end.

Jeff: It is.

Speaker 1: Go ahead. How would you approach that from your angle?

Jeff:

[00:24:30] It is a business issue, but it's also a security issue. There's a lot of people that say, "Our tool does everything except for business logic," and they kind of use that as an excuse. We can't check business logic because that's too hard. But really a lot of the business rules are the most interesting from a security perspective.

[00:25:00] I worked on a number of different mutual fund applications, where depending on what you were holding in your accounts drove the kinds of things that you could see. If you're over 65, then you can see options for retirement accounts that you can't see if you're under 65. There's all sorts of rules about how much value you have in your account, allows you to do certain other things and get certain benefits. The rules get really complicated.

That's just one challenge that application security has. It's difficult because it's all clay.

Speaker 1:

[00:25:30] Interesting. Where does, in the genesis of, I noticed from some of the patents that you developed, and now when you launched your company, maybe you can talk a little bit about where ... You obviously identified a need and a gap that the current capabilities of application security had, and I'd love for you to take us through, what was the defining moment that said, "I have to go solve this problem in a former company, turn my patents into products." Where did this happen for you?

Jeff:
[00:26:00] Actually, it grew a little bit out of our participation in OWASP. We developed a culture at Aspect of building things and making them open source. We built a lot of tools and documents and standards and contributed them, through OWASP, to the world. Some were small little things, like a tool to test for [inaudible 00:26:12] forgery. Others were more ambitious, like the enterprise security API project. We got really ... That was part of the culture at Aspect, is to build these things and try to make the world a better place.

[00:26:30] At the same time, we were doing work for lots of clients. We did security testing on many of the most critical platforms in the world. We were always looking for ways of doing that job better and faster, and we tried using static and dynamic tools to find vulnerabilities, but it wasn't worth it. Those tools generate so much noise, and at the end of the day they miss the important problems, sort of the false negatives issue, so we were always looking for better ways to solve that problem.

[00:27:00]

[00:27:30] In 2009, I did a paper for Black Hat. It's called Java Enterprise Root Kits. It was a paper of what a malicious developer ... It sort of studied what a malicious developer could really do inside a big financial organization, because nobody had ever really thought about it too hard, that I could find, and I thought it would be interesting to explore, so I wrote, it's like a 50 page paper, with tons of code in it, and lots of examples of ways that a malicious developer could put something into the code at a financial and hide their tracks.

[00:28:00] In there, one of the examples that I discovered was the Java instrumentation API. What it allows you to do is actually modify a running application, so you can change the code of a running application. If my thought it, if I'm a bad guy, that's fantastic. I could just connect to an app, change the code, have it do something malicious for me, and then delete itself, so there would be no traceability of my attack.

[00:28:30] I was like, that's pretty cool. That would be a neat attack. But in the back of my head I was like, wow, it seems like there ought to be something good we could do with that. I had this idea that we could instrument an application, basically put sensors inside the application, that would allow us to watch it run, and detect vulnerabilities better than we could do by either analyzing the code or the HDP traffic. Essentially this, by putting sensors inside the app, they would have access to much more information. Not only the code and the HDP requests, but also all the back end connections, the real data flow, the configuration files, everything.

The idea was, if we had more information, we could do a better job faster of finding vulnerabilities.

Speaker 1: Interesting. You're almost inserting a smart robot inside the app to watch for certain ... You're calling it an instrument, correct?

[00:29:00]
Jeff:

[00:29:30]
Well, instrumentation is a well known process. We've been doing it for 20 years. The idea is that you can instrument an application with some kind of sensors, and then those sensors will actually run as part of the application. Many people use instrumentation for things like logging. You might want to enhance your application so that it logs better, and you can use instrumentation to weave in that logging code. Many people use it for performance monitoring, so the instrumentation can measure how long certain transactions take. How long does a round trip request take to process? Where's my most expensive query to the database, so I can go optimize that? Tools like New Relic and App Dynamics, and Dynatrace, that's how they work. All we did was say, "Hey, you know what? We can use instrumentation for security."

[00:30:00]

[00:30:30] It's interesting. The performance market, in the old days, we used to have to hire some guy who was a performance expert, and he'd come in with a black bag full of special tools, and he'd go away for several months, and he'd come back with a PDF report with recommendations. Then we'd implement some of those, and stuff would get faster. But when New Relic and App Dynamics came out, those guys made it possible for anyone to do their own performance monitoring, using this instrumentation technique. They really democratized performance management. I think that's really what we need to do with application security. We need every developer. Even novice developers, we need to give them the capability to see their own vulnerabilities and fix them without involving an application security expert, because frankly, there aren't enough application security experts.

Speaker 1: You can't scale them. You can't scale them.

Jeff: As a planet, we're going to produce 111 billion new lines of code next year.

Speaker 1: Oh.

Jeff:
[00:31:00] There's no possible way that we've got enough experts to analyze all that code for vulnerabilities, so we're really ... We are pedaling backwards a little bit here.

Speaker 1:

[00:31:30] I see what you're saying. Back to our conversation about making security move at the pace of business, which is really the pace of software development, and innovation, your thesis is that we build these instrumentation ... I'm calling it from a layman's terms, like a basically like an exterminator robot built within the application itself, that can basically provide like a security sensor for changes in code, things of that nature, within the code, that would present a problem from a development point of view, from a security perspective. Is that correct?

Jeff:

[00:32:00]

[00:32:30] Yeah. Here's how that really works. When you take an instrumentation agent and you add it to an application, what it does is, it puts sensors all over the application, and it installs them into the code, as though they were written in the source code in the first place. We're just weaving in little sensors that allow us to snapshot what was going on at various security critical points all over the application. Then as the application runs, as the developer is building and testing it locally, or later on as you're running your regression test or your selenium test suite, or however you use the application, those sensors start firing as that code runs, and you can see what's going on inside the application. Because it's directly measuring the running application, it's a lot more accurate than if you're trying to guess what the application does by looking at source code or HDP traffic.

Speaker 1:

[00:33:00] How are you going to know that someone ... Are you loading at some level ... Are you saying, here's the known 300 types of coding errors that would result in obvious flagrant, like almost like the referrer saying, "Foul ball?" You've made a foul, basically, in the way you've coded this, and you just said to this instrument, "Listen, we want to," almost like a signature, you're saying, "These are the 300 signatures that are most coding errors, that can take care of 90% of the problems in a [flawed 00:33:14] application build?"

Jeff: The word signature is really a loaded word, and that's not the way that this works.

Speaker 1: Okay.

Jeff: I think of it more like behavioral guard rails.

Speaker 1: Okay.

[00:33:30]
Jeff:

[00:34:00]
If you take something like sequel injection. The definition of sequel injection is untrusted data flows through the application and ends up in a sequel query without having been properly escaped or parametrized. Right? That rule is exactly what we instrument into the application, so contrast tracks all the untrusted data, so any data that comes from an HDP request, or another untrusted sources, contrast tracks it as it flows through the application. If it gets properly escaped or parametrized, then contrast tracks that, and then when that data gets built into a sequel query and sent to the database, contrast can check and say, "Hey, did this path of execution, did it violate this behavioral rule that we set up?"

[00:34:30] With a single rule like that, we can detect a broad array of different kinds of sequel injections. Blind sequel injection, timing. There's all sorts of different kinds of problems that you can detect that way, so it's not like a signature like a virus checker where you can easily bypass it by changing a bite or a line in the code or something. It's detecting these patterns of execution that violate the rules. That's what we do.

[00:35:00]

[00:35:30] You can have either positive or negative rules. Contrast comes with a bunch of what we call negative rules. The application should never do this. But positive rules are actually more powerful, and if you can imagine you're an organization building code and you've got a secure coding guideline, maybe it says, "Every one of our spring controllers needs to call this access controller to do an access control check." Okay. That's a great rule. It's about two lines of XML to model that in Contrast, and then instantly, all your applications will start telling you, "Hey, anywhere you've got a web service that doesn't make an access control check, Contrast will report that right away."

Speaker 1:

[00:36:00] Interesting. If I ... I'm a business who doesn't have a ... I'm not a Fortune 500, but I'm still kind of an SMB, I'm 100 to 10,000 employees. I'm in that space. I maybe don't have a fully developed security team like Chase would have, for example. I have a business that's ... Let's say I've got several different applications that are in different hosting providers that the business has acquired, SAS type application. Is there a way to ... But I know the applications that I'm buying are being bought for my business benefit. They're not necessarily being secured the way I would expect them. Can I lay the Contrast, your application, or your techniques, into a third party applications?

[00:36:30]
Jeff:

[00:37:00]
Yeah. In most cases the answer is yes. The only requirement is that you have to be able to add our agent to the application. Can you go to Twitter and say, "Hey Twitter, will you add Contrast?" They probably won't do that. But if you ... If it's an application that you've bought and you're running internally on your network, or in the cloud, you can certainly do that. In some cases, if you're running on a platform as a service, you can add Contrast to those environments, and we're getting more and more integrations with those environments already.

[00:37:30] From the technology perspective, Contrast will run anywhere the application runs, so we don't care if it's in a container or a cloud or elastic environment or whatever, Contrast goes where the application goes. One of the things that's really neat about that is if you have an application that's internal, that you want to move to the cloud, when you move it, Contrast can go with it. We haven't even really talked about it yet, but the other half of Contrast, beyond the vulnerability detection piece, is the ability that Contrast brings to protect the application against attack. You started out a discussion, talked about WAFS. WAFS are kind of a primitive protection for web applications.

[00:38:00] With Contrast, we weave those defenses directly into the application, where they're close to the assets, where they can use all the context available, to only block real attacks. What that means is, you don't have to do all that tailoring and configuration that you have to do with a WAF.

Speaker 1: Okay. I think I understand that piece. Can you just explain that one more time? I definitely understand the WAFS approach, and then how is your approach different than a WAFS?

Jeff: Yeah. A WAF has to block attacks at the perimeter, right?

Speaker 1: Yes.

Jeff:
[00:38:30] When an HDP request comes in, it has to look at it, and it has to analyze the data there, and it has to say, "Oh, look. That's got a single tick in it and the word select and a less than sign," or whatever, "so I think that is sequel injection, or cross [inaudible 00:38:35] scripting," or whatever. But the problem is, you really can't do that there accurately, because you never know what the application is going to do with that data, right?

Speaker 1: Right.

Jeff: It could be something that just looks a lot like sequel injection, but really isn't. Then you block it, and then you've got to get people to come in and tailor the WAF and all that stuff.

[00:39:00]

[00:39:30] What Contrast does is different. Contrast receives that HDP request, it tracks the data from that HDP request, wherever it goes, and let's say a little chunk of that data ends up landing in a sequel query. Well, contrast analyzes it there and says, "Hey, did that untrusted data modify the meaning of this sequel query?" If it did, then we've confirmed the attack. We're not guessing. We actually watched it. We know that it's an attack, and we stop it there. Many of the ... Much of the data will never go anywhere near a sequel query, and so blocking it because it's sequel injection is kind of ludicrous, right? It doesn't even go to a sequel query. That's why Contrast is more accurate.

Speaker 1: You're not just watching it. You can take action on that.

Jeff: Yeah. Then we can prevent that query from going to the database, because we know it's an attack, so we block it there, close to the data. Ultimately that's it. Contrast has a lot more context than those other tools, and that's why it's more accurate.

[00:40:00]
Speaker 1:
I think one of the pieces we might have missed is that you launched Contrast, your business Contrast, when did you launch it?

Jeff:

[00:40:30]

[00:41:00] In 2013. We spent ... From 2009 to 2013, [inaudible 00:40:16] and I were doing tons of research. Nights and weekends. Trying to get our instrumentation platform built and really, really solid. We came up with tons of inventions along the way that became Contrast, and it's not the easiest thing to do in a world, to come up with a platform that allows you to instrument across all the different platforms that are out there. There's ... It's not just Oracle, JDK, running Tomcat. There's IBM JDK'is, there's all different containers, web sphere, Tomcat, Glass Fish, J Boss, everything. There's zillions of environments. A bunch of different operating systems. Bunch of different versions of Java.

Then we added dot net and no dot JS and we're currently in the process of building Contrast out for Ruby, which is in Alpha, and also Python.

Speaker 1: Oh, fantastic.

Jeff:

[00:41:30] Yeah. We're expanding quite ... Basically, our strategy was, we want to really prove out this instrumentation approach, make sure that it's viable, that it works, that it does better on every dimension than the old tools, and then expand out to lots of different languages, and that's ... We're in the process of doing that now.

Speaker 1: This is a billion dollar problem that you're solving. Someone said if you want to make a billion dollar company, solve problems for a billion people. This is a big fricking problem that you're trying to solve.

Jeff:
[00:42:00]

[00:42:30] It's interesting. The network security market ... The application security market is, it's probably about a billion dollars, if you add up everything. The network security market is like a hundred times that. But this is what's weird, is if you look at the Verizon DBIR and the other studies, most of the big breaches, by a wide margin, application attacks are the leading cause of breaches. It's like, twice as much as the next biggest cause of breaches. You read a lot about things, about different attacks in the news, viruses and worms and malware and so on, but really, application attacks are where companies are getting breached, and it's never ceased to surprise me that the spending doesn't match up.

That's I think a reflection on the failures that we had at OWASP, is we didn't ever figure out a way to solve the industry's problems in a way that they could scale.

[00:43:00]
Speaker 1:
I think it's kind of like the bullfighter. I think they're sort of letting that ... They're letting it pass, for most businesses. Again, unless you're Chase, and you're some of these really sophisticated security departments, and certainly they're not all that successful either, but I think people are letting that ... Because it's hard. That front end, the application firewall, and I'm using it as a metaphor now, essentially the protecting the app is very difficult work.

Jeff: It is. [crosstalk 00:43:28]

[00:43:30]
Speaker 1:

[00:44:00]
The instrumentation, that's a big problem to solve, which I think has really big ramifications, especially if you develop that, and we talk about Agile and rapid testing, and people moving towards ... We don't have one, two, three million dollar investment projects now in application development in the CIO groups that I run, that stuff's over. That's a job. That's a career killer these days. People have to be able to release rapidly, test, fail, pull back, test again, because the market tells you if you're winning, and the market can tell you very quickly, and unless your security can be elastic and fluid like that, it's going to be very difficult for a CIO in a [Sisso 00:44:13] to win.

Jeff:

[00:44:30] I was excited that Gardner chose us as the best security tool for Devops organizations, because we run in parallel across your entire application portfolio, all at once, we really have a much different scalability story than any other product on the market.

Other products will say, "Oh yeah, we're great for Devops, because we integrate into Jenkins," but when you really dig under the hood, it's really just automating the pushing of the scan button, and it doesn't really fix the problem. You still end up generating a giant pile of false positives that humans have to go through. You're just moving the bottleneck around. We really actually are working well in large organizations with thousands of developers.

[00:45:00]
Speaker 1:
I was going to ask you about, last year seemed to be a year of a lot of awards, not even counting the 2017 RSA, I saw you win the Sandbox Award, but you seem to be gathering the attention of Garner, for example, and others on your website. It wasn't like one or two. It was like seven awards last year, right?

Jeff:
[00:45:30] We've won a bunch of awards, but really the measure of our success is really how many applications we're protecting and whether we're moving the industry forward on actually achieving application security at scale. The vast majority of applications, nobody even does any security testing on. There's no protection in front of them. That's even in large financial organizations. If they're honest about what they're actually doing for app check, it's really not nearly what you'd expect.

[00:46:00]

[00:46:30] There's a weird disconnect in the market. I think consumers are surprised when someone gets hacked. They're like, "Oh my gosh. How is that possible?" But when you look at the data, the applications are, on average, you see 20-22 vulnerabilities per app, depending on which data you look at. The recent OS top ten did a data call from a bunch of different vendors. They gathered I think 2.3 million vulnerability across thousands, 50,000 applications, something like that, and on average 20.5 vulnerabilities per app. That's crazy. Can you imagine flying on an airplane where they said, "Well, there's 20.5 safety problems on this plane."

Speaker 1: That's the thing. Have you read the book Checklist Manifesto?

Jeff: Yeah.

Speaker 1:
[00:47:00]

[00:47:30] Checklist Manifesto is really interesting. They talked about three major industries. Healthcare, putting up complex buildings, and any building's complex, but putting up buildings and skyscrapers, and planes. It's funny. They used to lose a lot of planes in the early 1910-20s and 30s. Lost, as a percentage of flights, a lot of planes. Then some breakthroughs at Boeing and some gentlemen there that really worked on complexity theory and how to wring out a lot of complexity to make things a lot simpler for people flying and maintaining the planes then buildings, we don't lose buildings, but there's just a vast array of complexity to putting building up, and then we used to lose, we still probably do at some level, but used to lose a lot of people in surgery. The guy's point was that, I won't divulge too much about the book, but you have to reverse engineer complexity. You just can't keep accepting complexity.

[00:48:00] There's 15 people involved in an operating room. Why are we losing 300,000 people a year, in a Western world, with all the surgeries? Why are we losing people for line infections and all this stupid stuff? I thought, gosh, this is exactly where we are with security, and someone has to take a step back and say, are we going to dive into this complexity and figure it out, or are we going to let it dominate us? Where do you think we're going to go with this, Jeff?

Jeff:
[00:48:30]

[00:49:00] We need a lot more science in application security. The industry's kind of dominated by these giant maturity models, that somebody just wrote down a big list of everything everybody could possibly do for application security, and nobody knows if any of it works. They really don't, because nobody's done the science to test that kind of stuff. [inaudible 00:48:44] and I, we've got a ton of data now at Contrast labs that we're studying, from thousands and thousands of applications now, and we're studying the shit out of those applications to try to figure out why are they ending up vulnerable, and what can we do to help developers avoid those problems in the first place, because that's really ... The only way this scales is if we can get developers to find and fix their own vulnerabilities before they check in code. We just need some different techniques there.

Speaker 1:
[00:49:30] This is great. I think you have a unique approach to it, and I'm really happy you came on the program today and shared with us a bit of your expertise. As we get wrapped up, how can people reach out to you and learn more about you and your organization, Contrast, and I'm going to put summary of show notes together, so people can certainly click on the show notes, but is there a way that you, either through Twitter or LinkedIn or otherwise, that you'd like people to reach out to you?

[00:50:00]
Jeff:
Twitter, LinkedIn, and email are all fine. I should be pretty easy to find. At planet level on Twitter, and I should be fairly easy to find. You can also come to the Contrast website. There's a lot of information about what we're doing there, and my contact information's up there. I'd be happy to talk to you. Particularly if you're wrestling with application security in your organization, and you want to think about a way that's a little more modern, Devopsy kind of approach to app [stack 00:50:26].

[00:50:30]
Speaker 1:
Yeah. I love it, because it's a very different approach. I think it's a fresh approach that we need within the security community, and I want to do my part to get the word out of what you're doing, so I appreciate you for sharing all of your wisdom with my community.

Jeff: Thanks, Bill. Great to be on the show, and let me know if you ever want me to come back.

Speaker 1: Excellent Jeff. Thanks very much for your time today.

Jeff: All right. Bye now.