EPISODE 474

[INTRODUCTION]

[0:00:00.3] JM: When I log in to my bank account from my laptop, I first enter my banking password. Then the bank sends a txt message to my phone with a unique code and I enter that code into my computer to finish the login. This login process is called two-factor authentication. I'm proving my identity by entering my banking password; the first factor, and I'm validating that I am in control of my phone; the second factor, by receiving that text message. In order to login from my laptop, I need to be in control of my laptop, so the laptop itself is a factor. With the laptop and my password, I have two factors. I might not actually need the phone as a factor.

Praneet Sharma is the CEO of Keyless; a product that moves two-factor authentication into the browser. Praneet joins the show to discuss how all kinds of authentication work; multi-factor authentication, single sign-on and UBKey. We use this discussion of authentication methods to help explain why it might actually make sense for some people to be doing two-factor authentication without taking out their phone. We also explore recent security breaches like Target, Equifax and Yahoo, and the industry of security software that is sold to developers.

I see giant banners for security software companies every time I go into the San Francisco Airport and Praneet help explain to me some of the products that these kinds of companies are actually selling. Praneet has joined the show in a previous episode to talk about advertising fraud and he also works with my good friend, Shailin Dhar, at Method Media Intelligence. They are combating ad fraud, and this is not a paid spot but if you're interested in checking out Method Media Intelligence, they are looking to hire right now and I just love those guys, so you could email them, jobs@methodai.com.

Let's get on with this episode.

[SPONSOR MESSAGE]

[0:02:11.1] JM: So you've got a bacon delivery service and you need to notify your customers when their bacon has arrived at their doorstep. Twilio helps you make sure your customers get

the bacon while it's hot. Twilio's programmable API lets you build SMS of voice alerts easily in the programming language of your choice all in under five minutes with only a few lines of code. Now, your customers get a text or a call the instant their bacon is ready. If your customers want to see the bacon frying on a hot pan, Twilio has video APIs and SDKs for the platforms that you know and love. Learn more at go.twilio.com/podcast and get an additional $10 when you sign up and upgrade your account. That's go.twilio.com/podcast. you will only pay for what you use and it costs less than a penny to send a text.

Get started at go.twilio.com/podcast. Get your bacon delivery service cooking with Twilio's APIs for voice, SMS and video.

[INTERVIEW]

[0:03:29.9] JM: Praneet Sharma is the CEO of Keyless. Praneet, welcome to Software Engineering Daily.

[0:03:35.8] PS: Hey, Jeff. Thanks for having me.

[0:03:37.0] JM: Today we're talking about Keyless. We talked last time about some stuff in advertising fraud and hopefully later on we can catch up on how you're doing at Method Media Intelligence, but we're talking about your other company today, which is Keyless. I want to give people an educational walk through that leaves them to the motivation for you starting Keyless. Let's start with the simplest form of authentication, which is username and password. When I login with a username and password, what is happening on the client and what is happening on the server?

[0:04:14.5] PS: So what's happening with the username and password, you've probably noticed input fields within any sort of webpage, and what's happening is that input field is actually — That form is actually posting to a server and actually taking that string of characters and actually validating against the database. In its simplest form, that's what it is.

What actually happens on the database is the password is actually hashed, and what that means is to actually obscure to the actual people who control the database, so I can't just leak

database of passwords and steal everyone's password. That's one important notion to actually consider.

[0:04:52.3] JM: How do corporations manage the password data once I've given it to them?

[0:04:58.9] PS: That very much depends on how they choose to manage it. A lot of the issues actually stem from developers actually not considering what type of hashing algorithm they're actually using. Are they hashing it all? Which is very concerning, because everything is kind of in raw text. That's very much determined by how they choose to do that. There's no sort of transparency around like, "Hey, this is the type of hashing we use for our passwords." Have you ever kind of considered when you log in to Blue Shield or any sort of insurance website or let's say any sort of media website, there's no kind of indication on, "This is how we hash our passwords and it's secure," based on these requirements?

[0:05:40.3] JM: Yeah. Troy Hunt has published a number of reports on this companies that are just storing their passwords in plain text sometimes in places where there are some open ability to read those passwords back and it can be — There're just gaping vulnerabilities in some systems.

[0:06:06.0] PS: Absolutely. I think it really much stems from how they're stored, but also how that authentication systems works. If we're talking about just plain old usernames and passwords, the most common way of actually kind of breaking these systems, since it's just a doorway. Imagine, the analogy I use is just a lock and a key and there's a doorway to a house. Some smart systems actually try to limit the number of events, the number of failed events you have in authentication, but there are many occasions where not only are the passwords stored in raw text, but there are many cases where you can try as many times as you want, and that lends itself to automation. Wherever you can have automation especially in a doorway is where you can start kind of breaking these things open in a scalable fashion.

Then we kind of consider that a lot of people want memorable passwords, and memorable usually means simple. So they usually pick 123 password, ABC. Very, very weak kind of passwords that are easily kind of — They're crackable. Also, they pick passwords that are very much related to them. With some clever social engineering, if I knew your birthday, if I knew

your hometown, your dog's name, your mother's maiden name, these sort of features can actually narrow down what your password might be, and the search space for that kind of brute force attack goes down. It actually makes it way more efficient. If I can hit something thousands and thousands of time a second, the chance I'm actually going to break it is very, very high.

[0:07:42.8] JM: How are those brute force attacks implemented? Do they distribute it among a bunch of different IP addresses so that the server can't recognize that, "Oh, all of these logins, these dictionary attack logins, they're coming from a bunch of different places."

[0:08:01.3] PS: That could be the case, but sometimes you don't even need that. Again, we've had many conversations about ad tech and how they try to throttle IPs and people use botnets and proxies to actually have a diverse array about IPs.

Usually with some authentication pages, that really doesn't matter, and then you have to consider kind of the development teams that are actually controlling these environments. Is that something they want to annoy themselves with? Do they want to actually consider this and actually code that in nginx or their web server or their application architecture anyways? It really depends on how much they've thought about this.

With more sophisticated and more secure authentication gateways, that is kind of a measure they use, is IP throttling, right? You can send more than 10 requests per second from this IP. Very similar to how other bots or botnets work, except they're very focused on just like an input field. They're very, very simple, but usually what they do is iterate through just common strings that people use. If you look at your keyboard right now, the qwerty keyboard, human behavior is very, very simple. When it comes to just passwords, like we might just pick characters that are next to each other, so that makes it a little faster for the actual bot. Again, there're words, common English words that people might just use for passwords. Again, the search space gets limited from there. Then there are certain things, like proper nouns, like city names, all these other things that are actually incorporated.

Very rarely do you see someone that has a very, very obscure passwords, right? Like it's an obscure string of characters. Safari actually tries to generate some for you. They recommend passwords when you're first filling it out and you might notice that it makes no sense, it's

meaningless, but it's also very hard to memorize, so they store it in a keychain. Very rarely do you ever see humans just use that, because it's hard to remember. Our brains aren't configured that way. For a bot, the search space is actually quite small when it comes to common passwords.

[0:10:06.1] JM: When we authenticate with our username and password, we're demonstrating that we have the possession of a key, and most keys can be stolen or copied. We have two-factor authentication.

[0:10:23.8] PS: Yes.

[0:10:24.2] JM: How is two-factor authentication — I think most of the listeners know what two-factor authentication is. It's; I login to something from a strange IP address, so I get a text message sent to my phone and it verifies that I am in the possession of my phone. The phone is the second factor. First factor is the computer that I'm logging into, hence two-factor authentication. More generally, this is multifactor authentication. On a technical level, how is two-factor authentication implemented?

[0:10:58.8] PS: Great question. I think first and foremost it's good to go over just authentication and being something you know, being the username and password that we just discussed, something you have, which is a phone usually or like a key generator and something you are, which is a biometric signature. It could be fingerprint, now with face ID. There're also your eyes and your facial structure. Those are kind of like the three dimensions of this.

When it comes to implementation of MFA, there are different sort of scenarios. One, there might be just mandated MFA for every authentication event or every kind of transactional event that requires kind of a second factor. Think about a bank transaction that's of a certain amount. Your bank will actually send you a text message to confirm that amount for every transaction sometimes.

When it comes to authentication, multifactor authentication and its implementation, it's really just an extra step. You'll have the initial one where you'll check username and password, you'll check it again. Hopefully it's strongly hashed database or a stored database and you'll actually

check again, “That’s fine. That passes.” Then you’ll have a second step. It’s usually a redirect where it’s like, “Okay. We either send you a text message. Enter that code and we’ll check that code for this duration of time.”

One thing to note about multifactor authentication is that code isn’t really permanent. It expires. It’s usually 30 to 60 seconds. It could be text message. It could be an app on your phone that is generation this signature every 60 seconds based on some code, or it could be a key generator that you actually have on your keychain. You’ve seen the RSA tokens. Hopefully people have seen those. People lose those all the time, but those are actually generating those time-based six-digit codes that you can actually input as your second factor. That’s pretty much it. It’s just kind of an extra barrier.

From the implementation standpoint, it’s actually quite simple. You just hit another endpoint to validate against this for this user, for this current time span. It’s actually very simple. There are many services that I actually abstract this notion and allow you to actually build it into your web service and it actually strengthens security quite a bit. The annoying part is users have to actually configure it and a lot of users are lazy, but then a lot of users actually lose that second factor. It’s quite common. Authentication, the worse points in authentications is just in a business scenario where you experience a lot of lockout where someone cannot access their computer for 24 hours or more. There’re a lot of hoops and hurdles that kind of prove that you are you again. A lot of multifactor authentication scenarios involved. Just backups and resets and everything, and that’s where developers also have to kind of incorporate that into their web architecture.

Again, this can be all abstracted if you use a third-part service. Those are great, but there are instances where, again, being there are legacy providers that have actually done this themselves and they might have misconfigured it or they made it a giant pain for the actual user to kind of reconfigure multifactor authentication.

[0:14:13.2] JM: Those little RSA dongles that you attach your keychain and you press a button and it generates a new code that you can use as your second factor, I’ve worked at places where they make you have one of these little key dongles in order to login remotely. I don’t

understand why I can't just use my phone as the second factor. Is there a reason that they give you these dongles and they require you to have it to login remotely?

[0:14:42.9] PS: There's no real reason. Your phone can generate those codes. The algorithm that generates it is just — It's very similar. Maybe it's just a legacy mandate. Maybe they have a deal with this dongle provider. There're also politics when it comes to security. There are tons of politics when it comes to security. You don't need those. Your phone can serve as that generator.

You've seen cases where these SSOs have actually been introducing apps that will just do all that for you. One is they'll just push. They have push notification type functionality where they'll just kind of send it over instead of you having to manually type it in. They've made that a huge convenience except you do still see the mandates where here's a key generator. It might be because that's just the way things are, or there's a very strict BYOD policy. They don't want any sort of malicious app actually interfering with this push notification intercepting your code. There's a huge issue with just the android ecosystem especially — There're a lot of security vulnerabilities. If you put a whole key generator on your phone and then let's say that gets compromised and your password is really easy to crack, then there's no point in having multifactor authentication. Actually, it's worst from there. Those also might be reasons where the key fob is so simple. It's just a very high security scenario that they're trying to enforce.

[SPONSOR MESSAGE]

[0:16:21.0] JM: The process of troubleshooting bugs can be very tedious and inefficient for developers, especially are they are pushing more and more code to production. The unlucky developer assigned to bug duty make it bombarded with error alerts and spend hours figuring out which errors to address first. They might have to deal with logs to piece together what happened or even spend time reaching out to other engineers on their team for help.

Bugsnag improves the task of troubleshooting errors by making it more enjoyable and less time-consuming process. For example, when an error occurs, your team can get notified via Slack. See the diagnostic information related on the error and identify the developer who committed

the code. Bugsnag's integration with JIRA and other collaboration tools makes it easy to assign and track bugs as they're fixed.

We have a special offer for Software Engineering Daily listeners. Try all features free for 60 days at bugsnag.com/sedaily. Development teams can now iterate faster and improve software quality. To get started, go to www.bugsnag.com/sedaily. Get up and running in three minutes.

Airbnb, Lyft and Shopify all use Bugsnag to monitor application errors. Try all features free for 60 days at bugsnag.com/sedaily.

Thank you, Bugsnag, for being a sponsor of Software Engineering Daily. It's much appreciated.

[INTERVIEW CONTINUED]

[0:18:01.5] JM: The single sign-on, SSO acronym, that's referring to Facebook or Google single sign-on. There are some other providers we can explore. That's like where I use my Facebook or Google login credentials to login anywhere. I just have an app that I open up on my — Like I can have apps where I open it up on my phone and it just login with Facebook and then I click on login with Facebook. What's happening there during an SSO authentication?

[0:18:33.6] PS: That specific one, login with Facebook and Google and what's a OAuth. That's OAuth application. Yeah. That's transferable credentials for using one identity as another one. It's kind of similar to SSOs. SSOs actually use SAML. It's another protocol. It's very similar. There are certain providers that are identity providers and they can act on behalf of these services. What SSOs do is abstract the notion of identity where you can login to 60 apps with just one identity using SAML.

It's really convenient again because you don't have to remember all these different disparate passwords around everywhere and you can login to Google, Facebook, ADP, Salesforce, all these other things with just one set of credentials or one multifactor of solution. That's the SSO notion. It's similar to OAuth, but OAuth has just made it more convenient to login anywhere that is a third-party accessible app that connects to Facebook or Google or Twitter.

[0:19:41.6] JM: Is the OAuth notion, is that more of a consumer-domain, and then the SSO or the SAML — Is that we call it? Is it S-A-M-L?

[0:19:51.9] PS: Yeah.

[0:19:52.3] JM: Is that more of an enterprise notion?

[0:19:56.9] PS: Yes, exactly. The reason they built OAuth is just to spread, right? Like people don't have to keep signing up for things and also data sharing. OAuth allows — Expose all these other API endpoints that you can use together, like friends of friends from Facebook or like interest, or like from Google it could be your email and other things. OAuth is more for consumer adoption, but for businesses to get consumer's data very, very easily from these known providers where they spend a lot of time.

For SAML and SSOs, that's very much an enterprise notion, because there are many different apps that a business user might use, from Salesforce to Gmail. They might have other internal apps that jive and there are so many different apps that you have to kind of say, "Okay. Either we can have people just try to remember everything, and that's where things fall apart, or we can just kind of bundle everything into this one catalog. The SSO provider, like Okta, will abstract everything and you can just kind of click. You can authenticate using multifactor. It's very secure. Then you can just kind of click on an app and it will just kind of initiate this redirect and authenticate you on Salesforce of Gmail." That's more for the enterprise, because the IT department just doesn't want to deal with 60 different credentials for 60 different apps. It's by end-by-end problem for them. In the past they used to do this and there were just so much lockout and there's so many vulnerabilities that just popup everywhere.

[0:21:35.1] JM: You've mentioned that security can become very political with an enterprise organization. What do you mean by that?

[0:21:42.0] PS: Political as in just with anything. It could be the people. It could be the tools that you use. It's the tools that you are willing to buy. The thing about security is it can be very obscure. In any organization, you can have politics from sales to marketing, but you can have — With these departments, you have more of an understanding. There's more of a general

understanding of just human behavior and like this is the way people lack in this organization. You can grasp sort of the politics and why they occur in sales, marketing, even engineering.

When it comes to security, there're a lot of unknowns. There are lots of — If I say hashing or encryption and the way things work, there're a lot of jargon that kind of pops up. There's also the intimidation factor. People like talking about security, but then they don't want to start to defecting them and all these bad things could potentially popup. There's just something you don't want to talk about.

I remember working at companies where we would have trainings. It would be for half an hour, and people just dreaded these half an hour trainings, but moreover, they just didn't get any of it. They were just like, "What does this all mean? Why do we use the tools that we use?"

They are aware of all the beaches and all the threats and like just how much of an acute disaster of vulnerability can entail, but the thing is the security professionals didn't make it easy for the individuals that didn't know anything about cyber security in general. They would use the same jargon. It's more of like a fear-based kind of education. I know it's moving away from that to kind of be more welcoming to actually say, "Okay. This is security hygiene." But it's also kind of a well-guarded territory. Security people can be very territorial of like, "This is what I maintain and I'm an application security expert or I'm like just a general cyber security expert and this is my domain. I don't want anyone interfering with this." It's just kind of this security, internal security. No pun intended, but they want to feel about their domain.

The incorrect way people have been going on about it in the community is actually just intimidating the hell out of people saying, "This is going to happen to you, and this can happen to you. Phishing is a real disaster, and malware is everywhere." It's the same thing where people just kind of want to forget about it. That's just a normal human reaction to just be in denial, especially when it comes to technology. They'll just not think about it, "It will never happen to me." When it does happen, security experts come up with all these different products then. That's why it get political, is just these products are very obscure, but their obscurity sometimes make them — It makes them very "valuable" and they can charge a lot of money for it.

Sometimes you see organizations by endpoint security or this type of solution and they have no idea what they just bought. They're kind of wondering, "What does this actually do for us, and have we tested this? Does anyone here understand this?" Usually there're blank faces everywhere, or you have an internal security team that is so convinced that this is the future of security and it's unbreakable. There's nothing that's unbreakable in the realm of cyber security especially on the web with all its protocols. That's where it gets kind of political, is just the intimidation. People buy tools based on either hype or sometimes they know the people that build those tools, so it's kind of like the silent handshake, and just the general obscurity. There's just so much technicality to it. Things are constantly changing. Then finally, I think, it's just denial from the general population of just like not actually thinking about it until it's a disaster.

[0:25:26.3] JM: Whenever I'm walking through San Francisco Airport, I see these huge ads throughout the airport. Sometimes they're on digital billboards or they're just these huge posters selling to enterprises who need the security. More recently, they're selling not just the security but the security plus the AI, and I see these companies, what are they selling? Do you have any idea?

[0:25:57.6] PS: It's magic. It's magical. It will eventually work with enough data. Just provide us enough network information and it will eventually work. I've seen visualization, these ideal visualization at a conference of our AI security product and it will dynamically learn what vulnerabilities are surfacing in your intranet or your active directory, just your connected instances in your cloud provided that you spend a year configuring it and then ideally we want this clean data, and then ideally everyone knows how to use the tool and configure it and everyone follows proper security hygiene.

Those are the fine points that they actually didn't tell you, because that's a billboard, one, but it's all magic. There's no sensible way to actually implement that at the enterprise level. You're throwing one complex system at another complex system. It's bound to have failure points, and security is all about bolting down your failure points. So if you put in this AI security and all of a sudden the enterprise organizations will just kind of cleans up their hands and they're like, "All right. We don't have to actually think about security anymore because we have an AI that thinks about it for us."

They have their behavioral hygiene kind of goes down and all of a sudden they hire contractors and all of these third-party people. They kind of skirt the fine lines on just password hygiene and enforcing multifactor and all of a sudden one contractor has a vulnerability and a breach and either runs away with a lot of sensitive material or is that failure point that corrupts like a computer network. Those are just cases that happen, but it's usually when people just stop thinking about it and they don't enforce them. They don't take things seriously anymore or they don't make a kind of a habit.

When you stop educating your people around and say like, "Hey, here's our AI solution and here's the magic that will take care of everything. Use your grandma's name as your password now. It's fine." That can have a lot of kind of failure points that are just getting introduced, because people get lazy. When I see these types of things I'm like, "Well, it's kind of has this double effect of you're exploiting essentially lack of knowledge. You're going into these pitches. You're going into this meeting with the executives and you sell them AI. That's what they want to hear, because they don't want to think anymore." Then you also sell them this fortress that will just kind of absorb their lumbering enterprise architecture and just do it for them, and they don't look at defined points and actually consider sensibly like, "Look. As an enterprise, we're a very complex system and there are a lot of dynamics."

They never kind of question like, "Okay. How does this actually work and how are we actually going to implement this? What's the actual kind of implementation phase look like? Actually, how does this AI work? What models does it use and what types of data does it need? Is it even proper to consider it an AI? Is it just a general machine learning model?" People don't ask those questions. They kind of just take it in and absorb it. Their CSO said, "Great." Signs it off and all of a sudden they have this huge product that they have no idea what it is.

[0:29:11.2] JM: It's not all vaporware though. I mean I know — I've done a show with Pindrop Security, for example, and Pindrop Security does security for voice interfaces or phones or if you're operating a call center, they help to verify that the customer who is calling is who the customer is, and the product makes complete sense. I didn't know what Pindrop did until I studied it. Some of these other security companies, they can't all be vaporware.

[0:29:46.6] PS: I agree. Yeah. That's a really good case with them. I'm familiar with them. They're very specific on what they do and they can tell you how it works and they can actually demo it, but they're also very focused. Security needs focus. It doesn't need a platform that you just throw a blanket on something. They're very kind of precise in what they do. They can elucidate it. They can actually explain kind of the problem statement. They can elaborate on it and actually provide that solution.

That's a good thing. You cannot just bolt down everything with a platform, especially in the realm of cyber security. You have to have these point solutions that kind of come together, but also to be successful with Pindrop Security, people actually have to use it and be educated on it and know that this is a problem that might be affecting them. Correct? It's all about actually the awareness around it as well.

[0:30:36.8] JM: Yeah. How should a smart enterprise CIO or a CTO be evaluating this technology? What should they be looking for? Should they figure out the vulnerabilities in their organization first and then find point solutions to secure those individual areas? What is a general strategy if you're a CIO or a CTO for purchasing "security software"?

[0:31:07.6] PS: I would say get involved and understand the pain points of your business users, right? Like the people that work under you or work parallel to you or perpendicular to you. Understand their pain points and why they don't use the things that are necessary. There are many cases where security professionals just ignore the common person. Like not everyone is thinking about just bolting down their laptop, going incognito and proxying all their information very securely. People just don't think about those things. Understand where people might be ignorant and then kind of offer them solutions and good principles to actually mitigate those things. You can't just take — I don't think it's realistic to take 8 hours in someone's day and just reeducate them over and over about security and why it's important. You kind of have to introduce these hygienic things first, like, "Don't do this. Do this. If you see this, train yourself around it to be better, because as business user, this not only affects the business's security, which is very, very important, but also your personal security. Like if you are comprised, there's a high chance that your personal email might also be compromised is your life is pretty much compromised from there. It's very important, because they want to ensure that not only do you have a healthy business life, you also have a healthy personal life, so it impacts your business.

You have to kind of start from there. Understand who these users are. Very few times do I see security people actually hangout with regular old sales people or marketing people or other engineers, right? They're in their own little world and they think everyone is probably doing this and they build an internal bias. That's first, I think, what a CIO and a CTO has to do, is just like go to the marketing people. Ask them what have they thought about — Have they ever received a phishing email and what would they do and like just constantly test their behavior and understand when they just don't want to use the solutions that you think might really benefit them in a secure environment. They don't want to install and update antivirus software all the time. They just want it done for them.

There are many solutions that popup that actually offer like specific security. There're cases where the SSOs have actually done a lot. An SSO is a very good blanket solution just for authentication though. It mitigates a lot of the kind of password , kind of remembering all these passwords and usernames in different domains and is abstracting that, but also offered as a byproduct really good security. You have to train your users on just like why this is important and why it might be a little annoying to have multifactor authentication, but this is what benefits you get. You kind of have to elaborate on this over and over sometimes, but that's just for authentication. There are other realms of just like, "Okay. What information are you sharing with your customers?" If you're working for, let's say, a software as a service company that has access to sensitive, there's a lot of hygiene that comes from just like what you share and what you don't share and what you obfuscate and don't obfuscate. That's also something a CIO and a CTO needs to kind of understand and educate their general business population, I guess.

In this free e-book, you'll learn about JavaScript frameworks. You'll learn about software design patterns and you'll learn about the advantage of using frameworks with UI libraries, like Wijmo. You'll also learn about the basic history and the purposes of JavaScript's top frameworks. You'll also learn how to integrate a Wiljmo UI control in pure JavaScript and in some of the top JavaScript frameworks that we've already were talked about, like Angular, React, View and Knockout.

Wijmo's spec method allows you to determine which framework is best suited to your project based on your priorities. Whatever those priorities and your selections are, you can learn how to migrate to a new framework in this e-book. Best of all, this e-book is free. You can download your copy today to help choose a framework for your work at softwareengineeringdaily.com/grapecity.

Thanks to GrapeCity for being a new sponsor Software Engineering Daily, and you can check out that e-book at softwareengineeringdaily.com/grapecity.

[INTERVIEW CONTINUED]

[0:36:09.0] JM: In the high-profile security breaches that have happened recently, Target, Equifax, Yahoo come to mind. These companies were the unfortunate situation of starting a — They were companies that may not have started as "technology companies" or not modern technology companies, but they had to evolve in a timeframe where policies and strategies and security ideas were very nascent, and they built their infrastructure in that environment. It's very easy to pass judgment on Target and Equifax and Yahoo, Yahoo particular. I do somewhat pass judgment especially as a victim of all of these, right?

At the same time, I feel some sense of sympathy for them. They've got — Their company swells at a certain point and then they get brain-drain as technology world kind of leaves them behind, Amazon destroys them. What's the way to adjudicate these cases?

[0:37:19.2] PS: I think these cases you saw, people might have gotten lazy on the development side or they just didn't think about everything all at once. Again, these are complex systems. I've seen many enterprises with like 50 different internal apps stitched by active directory. They're

just one failure point, like just asking for it. They're just the way things have been done. There's kind of a tension, right? With the engineering cycles and kind of application security cycles. You want to be secure, but you don't want to be so secure where you're not actually developing anything? That's kind of like a hostile immunity all of a sudden.

There's this constant intention of like, "We need to iterate and we need to deploy apps as fast as possible, but we also need to make sure they're secure," and that's such a kind of tradeoff, because it might take quite a bit of time to actually pen test an application or actually run through a checklist and it's all these bureaucracy. Some people might actually just try to avoid that or sometimes you might it. This is a complex system and things are constantly changing. I know with the Equifax breach, there was a framework of [inaudible 0:38:29.3] that was not updated. How do you manage all these things?

If you look at most applications, there are so many third-party packages. If you look at just a website, there are so many third-party libraries. Things are constantly changing and updating. There might be a vulnerability that pops up in a runtime tomorrow and you like you have to go pick up your kids at school. There are so many things within life that just like come and conflict with just thinking about security all the time.

With these organizations, like a Target or Equifax, they might not have the engineering resources or they may not have the internal resource that prioritize that. There are other things they had to prioritize, and all of a sudden they had a really acute disaster. Just like any natural disaster, there's probably places where they built up according to their economy and things that prioritize and grow their economy, but all of a sudden a disaster comes through and how do you account for these things in a very short amount of time?

Cyber securities that evolved, because things can happen overnight. Just one couple of minutes, everything could be just compromised. You could have a leak of raw text passwords. Two billion rows to a computer is not that much actually. It's very easy to kind of just run away with information.

I sympathize with these companies, because it might have been the last thing on their mind and then they get blamed for it because it should have been the first thing on their mind, but, "Oh!

You also have to build good products and you also have to prioritize your growth, your year-by-year growth and quarter-over-quarter growth and you also have to do these other things. If security gets in the way of that, then it shouldn't, right? It's just kind of a misunderstanding of how complex a modern enterprise is now.

[0:40:25.3] JM: Now that we have painted a picture for that modern enterprise, you started a company earlier this year called Keyless. Explain what Keyless does.

[0:40:35.1] PS: Keyless is kind of a reaction. It more have started as a reaction to just the world of authentication and it was me, while I was at New Relic actually talking to people that are in sales and marketing and in a very non-technical role, some of the engineering team up in Portland just around what do you hate about authentication. I know it's a stupid, simple thing, but it's the doorway to everything. It's the first point of entry. That door could be really secure or too secure where there were many cases where people were locked out of their machine. They couldn't access things. They wasted a day or a week or they're abroad and they probably got nothing done because they were just completely locked out.

I picked a really specific problem in security, cyber security. It wasn't just the entire platform of sorts, and I looked at the doorway where people have kind of been adding more locks to the door. They've been adding the multifactor, just like a deadbolt. They've been adding other things via SMS or push notification. There are many ways to increase security to that doorway, but what happens when you lose or forget that key? There's many instances where password reset is actually one of the biggest vulnerabilities where every day I actually — This is funny. I get an email from my WordPress site asking me to reset my password, because someone asked for a password reset.

Now, if I happen to click on that link, I would authorize that reset and things fall apart from there. Many times an actual reset is actually the vulnerability.

[0:42:07.9] JM: Why is that a vulnerability? Because then you are presented with the dialogue to reset the password?

[0:42:14.3] PS: Yes, but I would authorize that sort of unique code. Many password resets actually have a token that actually say, “Okay. If you click on this link, the internal server will actually say, “Okay. Time to reset the password,” and that links becomes authorized to actually reset it.” So someone can go in and say, “Okay. Now, time to use my name as the password and now I’m locked out of my thing.” Worst, someone else has access to my WordPress site, because they just reset their password to something they want.

[0:42:40.8] JM: The person who is resetting the password, they get the token identifier even though you get the email? Do they have access to that token?

[0:42:51.9] PS: The token is actually sent back as a response from the WordPress server. So they know what that is. Yeah. It’s archaic, but this happens. The other point of authentication is especially with the multifactor situations, people were losing their phones and people were losing their key dongles, and they were locked out just indefinitely and it was really annoying and trying to reprove to everyone that this is me. This is kind of — This is my identity. I need to kind of reconfigure everything so the actual SSO trusts me again.

There was a huge pain point. I thought about it as an engineer, I was like, “Why? It’s not that bad.” It’s just you lost your phone. You got to do this,” but for some people, like this really disturbed their lives. They hated doing it. There are cases where people would try to avoid the SSO. They would try to just go to the domain and see if they could just still login with their old credentials. There are certain domains, like SalesForce where you could do that with the way it was configured. It just was really concerning.

I kind of went into their shoes and I was like, “Well, what do you hate about the situation you’re in right now, this authentication situation? I know it’s dumb, again.” But they were like, “I just hate having to multifactor. I want to just kind of go in and just have the multifactor just handle itself.” That’s where I thought of like Keyless. I was like, “You can do this with your car. You have keyless entry.” Getting a little more technical, if anyone’s familiar with UBKeys, right? Those little USB keys. They actually use a really strong protocol called U2F. That’s kind of a handshake that an application can do on the web with this hardware device.

It's also an open protocol. It's well documented. So I took this protocol and I made what I call — It's a Chrome extension, the Keyless extension that works with Gmail, works with Facebook, works with Octa, and it does that handshake for you. It was just kind of a simple reaction to just like, "Does this make your life easier?" It did. For a lot of people they're just like, "Yeah. I have keyless entry and my laptop is kind of like that device. My work laptop is that device that provides that extra security." It's kind of is that second factor.

Now, people argue that, "Isn't that point of multifactor. So like what if you're laptop gets compromised?" Well, that's another problem. That's something that — Again, my philosophy on security is pick one really direct problem and solve that. Then other problems, there's going to be other solutions that solve that.

With laptop security, I mean if your laptop is compromised, if you have a malicious app on your laptop, I can just wait for you to login and hijack your session. I can just wait for you to do that as a malicious app. Pretty much, I have ownership of your identify from there.

Now, what you have to probably consider then is how do you stop that. That's not something Keyless will do. That's something that many other solutions will do, but there solutions out there, antivirus software. There are a lot of solutions out there that will help mitigate that. But there're also hygienic things that will mitigate that. Don't download that Flash DMG that was triggered by a redirect. Don't do that. There are a lot of programs, games or anything that you might just want to consider, like what are they using in the background. Maybe ask someone. Maybe ask a security professional.

It also takes the inverse. For regular people that comfortably go up to these security professionals and say like, "Should I download this random game off the internet? Could it be bad? Could it have a key logger? Could it inject some sort of other malware? What could it be doing?" These are questions people should be asking, people that are in the security industry, and they can freely ask them.

Again, it's considering kind of other — Cyber security is a realm of problems. You can really solve one problem well and then you can rely on a cohesive set of solutions to solve the complete umbrella of problems. If someone stole my laptop and said, "Okay. Your operating

system password is really weak." Well, that's not something Keyless was meant to solve. That's just my hygiene.

[0:47:12.5] JM: To clarify for people, you're basically giving people the ability to login in a typical situation where they have two-factor authenticate from — If they're logging in via browser, you're giving them the ability to skip that second factor. The reason you're doing that is because most of the bad actors who would be trying to login would be logging in remotely. If you install this browser extension on your specific browser instance, you use the UBKey protocol, so you are essentially automating the 2FA only from your browser when you're logged in to your operating system so that you specifically don't have to go through the pain of 2FA on any — This could be applicable to any of the times where you have to use 2FA on your browser, which for me is like two or three times a day probably, and there's a lot of wasted time and it definitely adds up to a lot of wasted time, because the person who's hacking you, they are not going to have the browser extension, or if they do have the browser extension, they're not going to have it authenticated. It makes complete sense to me.

Are there any added vulnerabilities? I guess it just is if somebody has your operating system password, they can login to you operating system. I guess that's the main vulnerability. What were some of the engineering challenges in building Keyless?

[0:48:47.5] PS: The engineering challenges are really just around working with the U2F library, the JavaScript library, understanding that and actually understanding how the handshake is consumed. I want to add that U2F has a very strong protocol. It's stronger than that six-digit token that you get. It's remarkably stronger than that, and there are cases where that six-digit token, it's vulnerable.

[0:49:12.9] JM: By the way, the six-digit token you're talking about, when I do 2FA and Twilio sends me some little — It's not Twilio. Whatever a 2FA authenticator provider is. It sends me like, "Login with 253496," and just like that. You're saying that that is more vulnerable. That is less safe than the UB —

[0:49:36.5] PS: Yeah. U2F is just much a stronger handshake. Yeah. that has cases where it's been compromised, and that's where UBKey, with their protocol, is really kind of advancing the

frontier. Like that USB key that they give you that you can plugin in a laptop or any laptop is just — It will authenticate with that handshake.

Now, again, Keyless is meant to be built as — It's an extension that sits in your browser so your device will be the authenticator. Again, the reason I built this was because I had a UBKey and I lost it and it's very small. It's the size of like half your pinky. Again, the problem with the doorway is if you lose one of those keys, you're locked out. I spent $50 on a UBKey and I had one for Gmail. I was using it, and I lost it, so I was locked up.

Reproving myself was quite easy because I had backup authentication, but it was just so annoying, because $50 just kind of just disappeared, one, but then I thought about it. I was like, "What is if other people have the same sort of — Something happened to them where they just lose it or someone steals it." This is something you have being an authentication measure. It's very easy to steal a small device versus a big device.

Stealing a laptop is more difficult than stealing your phone or your little UBKey. I was like, "Why shouldn't the actual laptop itself have the ability?" Your work laptop should have the ability to be something you have and you get two byproducts, it's faster, because the handshake can happen in an automated way and you're not worried about losing this other factor all of a sudden. You're not going to lose your UBKey and all of a sudden you're locked out, because those things are small.

I give them a lot of credit for like what they've built and they're very secure and everything, but those things are just so damn small that I just lost it. I think it was on a BART train, and that was concerning because now someone else has my UBKey and they'd have to kind of correlate it to maybe my identity and everything, but think about this in work situations where let's say I'm a contractor working on a pharmaceutical company and let's say one of the project managers is using a UBKey authenticating everything and it's a very simple password, but a UBKey. All I have to do is steal that UBKey and hit — I can just unplug it from his laptop when he's not looking, and from there I can just authenticate and that's where things will fall apart.

Something you have really has to be robust as well. Something you are, the biometric signatures are very hard to mimic. There are extreme cases where someone took a fingerprint

and actually can authenticate somewhere else. It's rare. I know there are cases of it and it makes a great story, but the way MFA has evolved is we're relying on phones and keys or dongles. Those things are — They're easily stolen, let's be honest.

[0:52:50.4] JM: Agreed. Who are you selling this to? Is this to the consumer or to enterprise?

[0:52:55.0] PS: They Keyless extension is free. It's for everyone, and that's also as a reaction to security community, because the common person needs to be secure. The Keyless extension, it's a very simple handshake. I don't know the people that install it, because that would be contrary to having security, right? It's just, "Go ahead and install it. It's free. It works on —" If you're going to authenticate one thing, just do it with Gmail. Your email is very sensitive and it is kind of that main failure point. There's also your social media account, Facebook, which could be another one. The extension is just out for everyone.

What we have been doing though is selling to enterprises with their SSO. If they us Okta, which we partnered with Okta very closely, and we're trying to bring enterprises out of active directory. The reason being is the build a lot of internal apps that connect to active directory, and they have a lot of moving parts when it comes to personnel. They have — Let's say one of the part organization shift to another part of the organization within a Fortune 100, or they bring in contractors and consultants. That's usually what happens with these failure points is someone who's transient just comes in and leaves and either they maliciously do something or most of the time it's accidental. They get comprised accidentally. They happen to work with this Fortune 100 at some point and, again, it starts falling apart from there.

The actual systemic kind of solution, we've been bundling it with an SSO saying, "Look. We understand behaviorally people are going to hate using MFA. We have solved this problem. The convenient thing about us is we can also enforce a device."

A really big problem in the enterprise is controlling BYOD, because the thing is people bring in their phones. Sometimes they're on laptops, they install software. They authorize other sort of web services on that browser of their personal device.

With Keyless, there's been — We've had a lot of traction when it comes to actually helping enterprises with controlling devices and endpoints, and that just is a byproduct of just living in that device. With a UBKey or another dongle, you can actually authenticate many other devices. It's just plain simple. I can have one UBKey authenticate multiple laptops.

A lot of the enterprise organizations are like, "We don't want that. This is your work laptop and this is what it will do and this is your only work laptop. How do we enforce that?" We've really been helping a couple of Fortune 100 actually implement this, and that is beyond the SSO. That's more at the session level. That's more kind of like a custom sort of agreement, and that's where our revenue stream is coming from. For the consumer adoption, I hope people just use something. You don't have to use Keyless. Use some sort of solution that actually protects one of your important services, whether it'd be Gmail, Facebook. It doesn't matter. Really adopt it.

I recommend Keyless because it just makes it easy. You don't have to think about it. In the very least, use like an Authy. I have Authy for other services that don't have U2F. U2F, it's a very strong protocol, but it hasn't been widely adopted yet. I hope it gets widely adopted especially by the banks, by the other services, by cloud providers, like AWS. I'd really appreciate that, because it's just so much stronger. In any case, just use another factor, because I bet many — The majority of people have simple passwords and just to remember and it's comprised one time or the other.

[0:56:44.8] JM: You also run Method MI with Shailin and it's been about a year since I spoke to Shailin for the first time about his plans to talk more publicly about ad fraud. Since in that last year, you and he have made really significant progress on combating ad fraud. It's been really fun to watch and interviewing you guys and covering you guys on a topic that does not receive enough coverage. It's been really fun.

It's also interesting to see that you are actually running two businesses at once. Tell me about the pros and cons of running both method AI and Keyless.

[0:57:31.8] PS: Yeah. Great question. I think Method has been more the involvement, just because just ad tech is so big. I'm not saying like cyber security isn't bit either. They're both very

big industries. There's been a lot of kind of immediately solvable problems in ad tech and we have a lot of clients coming in from Method MI. That's the bread and butter kind of scenario.

Keyless actually started way before Method. It was started as a side project, and the only thing I've been doing with Keyless. Again, this is kind of — Again, my philosophy on security is to solve that one problem really well, and it happens to be simple.

I haven't had to put so much time commitment into Keyless as much, because I built a solution, and that solution helps these customers and these individuals very well, and that's that. Where that might go in the future, I don't know, because, one; Method has a lot of business. We're getting a lot of traction. There are a lot of problems to solve in ad tech, not only from the human side but also the technological side. It's kind of this interesting mix, because ad tech and cyber security are kind of merging together. A lot of issues actually stem from the ad tech stack, where you have third-party JavaScript, you have third-part ads that are just loading everywhere. Now, you have cases where there's something called cryto-jacking, where you can load a third-party script that starts mining cryptocurrency using your host device. There're a lot of issues surfacing within the ad tech stack.

A lot of malware is actually distributed through advertising. A lot of the key loggers, a lot of the malicious apps are actually distributed through malicious advertising. It's a good kind of like, I guess, synthesis of these two worlds. With my background in Keyless and with my background in monitoring and learning everything about ad tech, it's kind of this good kind of collection and multimodal thinking you can actually enable.

When it comes to the future of Keyless, I don't know. I think it's been a very kind of direct sell, right? Like we work with an SSO. I didn't go out and build an SSO, because that would take way too much time. I found a really good partner in Okta and I was like, "Here is my authentication. Here is my factor. You bundle this in with your SSO and that's that."

They are very gracious about it and they're like, "Yeah, we kind of love this, and this is going to solve these sets of problems for our sets of our customers in this domain." It's not like an overarching like, "Everyone's got to move to Keyless." That's not going to happen.

It's been very kind of — I guess it's been very manageable, because I know people that have two or three companies, and it can be overwhelming because they're trying to build a platform here and a platform there. From a business and operational perspective, I look at Method as just like, "We're going to need a lot of resources to solve these diverse sets of problems, and it's my day-to-day."

Keyless is more just like, "Hey, I have decided that this is a going to be a channel sell and these are the things we do and nothing more. There's no more consulting service that we're offering, although I'd love to just give free educational talks." I don't know. This is what Keyless does, it's an extension, it works with Okta, it works Gmail, Facebook, etc., and that's that. It's kind of just like a period on it.

It could evolve, but who knows? I'm always going to kind of consider like how much time and investment do I want to make and also have like a social life and everything. That's really kind of my kind of being situated in the Bay Area, growing up in the Bay Area, having a family that are entrepreneurs and really being realistic about, "Okay. If you want to do this and this, can you modulate it in a way where you can give a manageable time commitment?" The way you do that is just from the operational and business process side. How are you going to sell it? Are you going to actively go and get on sales calls and demo it and are you going to keep building on it and build bells and whistles? No. That's just a decision I made and I think it's a really good decision, because I've never seen someone built to operational businesses and then be happy.

[1:01:58.9] JM: Yeah, I completely hear where you're coming from. I think you're right. Okay, well, Praneet, continued success with Keyless, continued success with Method. I'm sure we'll be talking again in the future.

[1:02:11.6] PS: Absolutely. Thank you so much.

[END OF INTERVIEW]

stuck, your business intelligence tools are choking on data. The problem could be with how you are managing your Redshift cluster.

Intermix.io gives you the tools that you need to analyze your Amazon Redshift performance and improve the tool chain of everyone downstream from your data warehouse. The team at Intermix has seen so many Redshift clusters that they are confidence that they can solve whatever performance issues you are having.

Go to intermix.io/sedaily to get a 30-day free trial of Intermix. Intermix.io gives you performance analytics for Amazon Redshift. Intermix collects all your Redshift logs and makes it easy to figure out what's wrong so that you can take action all in a nice intuitive dashboard. The alternative is doing that yourself, running a bunch of scripts to get your diagnostic data and then figuring out how to visualize and manage it. What a nightmare and a waste of time.

Intermix is used by Postmates, Typeform, Udemy and other data teams who need insights into their Redshift cluster.

Go to intermix.io/sedaily to try out your free 30-day trial of Intermix and get your Redshift cluster under better analytics.

Thanks to Intermix for being a new sponsor of Software Engineering Daily.

[END]