#169 - MFA Mishaps

[00:00:00]

[00:00:00] **G Mark Hardy:** Hello and welcome to another episode of CISO Tradecraft, the podcast that provides you with the information, knowledge, and wisdom to be a more effective cybersecurity leader. My name is G Mark Hardy. I'm your host for today, and we're going to be talking about multi factor authentication mishaps, or MFA mishaps for short.

If you're following us on LinkedIn, great. If not, please go ahead and do so. Subscribe to your favorite podcast channel. And please subscribe to our YouTube page because you get to get this plus a whole lot more. Now, before we get into the MFA mishaps, let's go through a quick review of multi factor authentication.

And back on episode number 74 called Pass the Passwords, we went over eight different types of authentication methods that were listed in Daniel [00:01:00] Meisler's CASMM, or the Consumer Authentication Strength Maturity Model. Now going from worst to best, we had at layer one, shared password. Use the same password multiple places across the internet.

Yeah, right, but it happens. Number two, unique passwords. But they might be short, they might be simple, and they might contain personal information that could be guessed. Number three, quality passwords. They're not just unique, but they're long, random, and they include special characters. But there's still password dumps and cracking that could be used against something like that.

Now, if you go to a password manager at level four, in addition to having unique passwords, you also store them securely in an encrypted archive. Well, there's account reset or takeover, and we've seen Incidents with at least one of these companies, and I'm not going to mention them by name, that at least for one of my clients, we decided to move away from them based upon some of our concerns about the liabilities and the password manager.

Well, that's level four. Level five would be the SMS based two factor authentication codes. Hey, you get [00:02:00] something that comes in on your telephone, and in addition to multi factor authentication codes, on the phone, you get the sense that, Hey, it's got to be your phone. Well, of course it could be phished and we'll talk about that.

How someone could actually get your code that way by convincing you to give it to them, but also SIM swapping, where somebody calls up socially engineers, a person at the telephone company and says, Hey, my name is G Mark and I just got a new phone. I need to have a new SIM, blah, blah, blah, blah, blah.

Okay. So it's better, but it's still not perfect. Number six, you have app based. Two factor authentication codes. And so now it's generated by an application like the Google Authenticator, the Microsoft app. So now when it's requires a code, I have to go ahead and consult my phone. Click, click, click, click, click, click.

But again, phishing or even potentially malware could compromise that. Now we're at layer seven, where we have an. app based codeless two factor authentication. And now it's going to prompt you to accept or deny an authentication attempt. And Microsoft has moved from beyond that just accept or deny to be able to go ahead and actually push out a two digit code, which you have to [00:03:00] then reflect in here.

So you gotta be looking at it and seeing it here and it'll give you a map. So if you say, yeah, this is not where I'm at. Maybe the person will say no, but again. There's some ways that you can spoof people, and I'll tell you about that. And finally, at layer eight, if you will, the pass less. In addition to managed passwords, your two factor authentication comes from some physical token or some built in trust center.

And now you have a much higher level of availability. But there's still a possibility of things going wrong. And so that's, we're going to talk about a little bit about how could things go wrong. The key takeaway from the episode that we did way back when, when you take a look again at episode number 74, pass the passwords, was that passwords and multi factor authentication methods are not considered equal.

In terms of security. Now, granted, there's going to be a risk with any authentication method, even passwordless. Well, I'm not convinced now. Last one, one of my coworkers dropped a client laptop protected by a YubiKey and the key itself, well, it snapped in two. And so [00:04:00] he walked into my office, because a couple hours you had been locked out of the client machine.

And you have to FedEx it back to get reconfigured for the security settings. And they have a backup. So he said, G Mark, do you have a YubiKey? And coincidentally, I had just ordered. Two of them. And so I pulled one on my backpack, still in the wrapper, and said, here you go. So that was kind of a just in time good luck.

He was able to initialize that new key while still authenticated. And my C and CISO became a cape for being the hero for that day. Now note that our client in this case had opted for level eight in this authentication strength maturity model. That's kind of the top of the line. Not so bad, but it got me thinking, well, why don't we roll out YubiKeys to all of our users?

Well, this time, instead of using the kinds that could snap off, there's little stubby ones they call a nano. So there's nothing to break. Then I got to thinking that, you know, all Windows 11 machines require a trusted platform module or a TPM. And so that's why if you check for updates on an older Windows 10 box, you'll get a message.

This PC doesn't currently meet the [00:05:00] minimum system requirements to run Windows 11 and therefore they want you to go ahead and buy more, bigger PC. Now usually it's the lack of a TPM that's going to trigger this alert. Now Microsoft has announced that Windows 10 will reach end of support on October 14th, 2025.

with their current version, 22H2, being the final version. Now, unless you think Microsoft is moving too quickly, their support for Microsoft Windows 10 will have lasted over a decade. Now, note the customers who subscribe to the long term servicing channel release, which will update every three years, not every six months, may have a longer runway.

But my research has shown that the LTSC, as it's called, is not common in enterprise deployments because of the fixed but, well, reduced functionality. I remember up until just last year when I would go to visit my Uncle Ron, that they still had Windows XP devices that were running because it's kind of like a medical thing where they keep track of stuff.

Doesn't communicate to the internet, doesn't go browse and things such as that. So sometimes there are situations where you can maintain a system, [00:06:00] particularly like in medical, where you let it run a long time. And if it works, don't fix it. Well, we're not going to go to that. We're looking at enterprise solutions, such as which most likely you're dealing with.

So let's go back to the question about using a TPM for authentication. It seems that if you have a TPM in every device, why don't you use that to authenticate? And bing, you're at level eight. But wait, not so fast, because there is a difference between a Trusted Platform Module, this TPM, and a Hardware Security Module, or HSM.

And therein, I believe, lies the answer. Now, if we take a look, a specialized hardware chip known as a TPM, Trusted Platform Module, is embedded into a computer's motherboard to store cryptographic keys used for encryption securely. Okay? And the TPM offers secure storage for cryptographic keys and certificates.

Allows for secure boot, can generate cryptographic keys, and even provide remote attestation to a system's integrity. Well, so far so good. On the other hand, a hardware security module is the device [00:07:00] that you can add to a system to manage, generate, and securely store cryptographic keys. An HSM can manage cryptographic keys throughout their lifecycle, can do hardware encryption, that is go faster, and has tamper resistant hardware to resist physical attacks.

So, now I'm learning that the difference, primary difference between a TPM and HSM is that a TPM focuses primarily on security of the platform. and ensuring system integrity. Whereas a hardware security module, or HSM, safeguards cryptographic keys and offers hardware based protection. And it's that latter that we want to achieve this Level 8 on our Consumer Authentication Strength Maturity Model.

So at the end of the day, just buying a new laptop can help you secure your hard drive, but doesn't offer the type of validation. an authentication we would want from an HSM. Now, if you know of a manufacturer that builds in an HSM at the factory, please send me a note on LinkedIn and I want to go ahead and update my notes.

[00:08:00] If you want to experiment with hardware security modules and you also have the appropriate rights and permissions, you can enable FIDO Security Key in Azure. Go to the Home, Security, Authentication Methods. Policies menu. A little box you can click and turn on. Now note that Microsoft states, quote, FIDO2 security keys are a phishing resistant, standards based, passwordless authentication method available from a variety of vendors.

That's a mouthful. But FIDO2 keys are not usable in the self service password reset flow. Think about that if you're using the SSPR. Now, by the way, for your 500 point Hacker Jeopardy question, what does FIDO stand for? It's Fast Identity Online. And the standards are at the FIDO Alliance webpage if you want to dig into the details.

All these references you'll find in our show notes. Now, usually, HSMs in the form of FIDO security keys are reserved for enterprises that have the money to buy more stuff and are willing to configure and manage their systems to use them. [00:09:00] But what about the rest of us who don't have the budget or the staff or even a mandate to do so?

Do we have to? Settle for something else that perhaps is less robust, but it might meet our risk tolerance. Now, if you start by choosing an inferior authentication method that does not meet your risk tolerance, you're potentially placing your organization in harm's way.

So, for example, It's not uncommon to see email used as a second factor for authentication in websites. The user experience looks something like this. First, you log into a website with your username. It says, please provide the secure code that we just sent to the email on file. You then check your email and copy paste or provide the code proving that you have access to the email address.

Then the website asks you to provide the password to the site to log in and after that, bing, you're in. And that seemed like a pretty good okay approach at first glance. The website captures a username and a password, which would be the first factor, something you know, as well as proof that [00:10:00] someone has access to the email box, which is the second factor.

So technically two factors, something I know and something I have, which is access to the email box. What could possibly go wrong? Well, one example I remember is when DFAS rolled that out, Defense Finance and Accounting Service. So if you're in the military, you know what DFAS is because that's where you get your paycheck.

And I remember when they turned that on, they would have an MFA code. And they said, sign up for this. You get an MFA code and it was good for 10 minutes. But it took 20 minutes to send. And so in that particular case, if you signed up for the telephone version, cause you thought that was better, you realize that, yeah, no, this isn't helping because I can't get in.

And I kept trying for weeks. Finally, I think I had to make a help desk call or I got lucky and it came in with like a couple of seconds on the clock, but make sure your systems are responding fast enough if you're going to time out. These authentication tokens, because otherwise you're going to lock out the users.

GoDaddy did that when they first set things up. I got locked out because I signed up for their MFA and it would take them longer to send the MFA

[00:11:00] credential than it was for the duration of that MFA credential. And for a few days, I couldn't go in there and update my website. Eventually they fix that, but you don't really want to create a denial of service attack on your users under the guise of security.

Cause then it's really kind of working at cross purposes to what you want to do. And as you know, as the listeners of CISO Tradecraft, we say that security is in the business of revenue protection, not reduction. So let's take a look at how an attacker might bypass an MFA email solution. A bad actor might first trick a customer into providing their email address username and password to their email box by creating a fake login page that looks like Yahoo or Gmail or whatever they're using,

so, for example, they'll phish the user by sending a text message saying, no, this is. Gmail. We've recently detected someone trying to log into your account from an unknown IP address. And if this wasn't you, please log in to change your password. Okay, well, there you go.

You're going to log into their fake site. And now they, once they've stolen your email account credentials, they can log in and start reading your emails. [00:12:00] And they might see an email from a company that says, for example, here's a core credit. Please use it before it expires. Or buy this or an invoice for some other site or some Amazon receipt, whatever.

So the attacker goes to your company website, but realizes you don't know the username and the password. Does that mean that if our customer's email addresses are compromised or MFA email solution remains secure from bad actors? Well, not necessarily because it's common for a company or a merchant website to have a forgot my password option.

We only need to provide an email address to get a link that you can click on to reset your password. And if that's the case, the bad actor will find that link and click reset my password. And now the bad actor controls the email address. They're going to choose a new password, which A locks you out. But then they can spend those unused store credits, order merchandise, ship them to a drop location, or do any number of possibilities, because they're, well, they're in as you.

And they're not going to necessarily trigger any alarms that are going to say suspicious user because, well, [00:13:00] you're a legitimate user and you're kind of doing what you would normally be doing. buying stuff or, or making transactions or et cetera. Now the other thing is that to make sure that you as a regular user, because sometimes they don't completely reset everything, they'll delete the change password email from the inbox.

So you don't even know what happened because from time to time, you might have your regular box on your laptop and also on your cell phone. You can check your email. If you don't see anything that would make you suspicious when it comes to periodic reauthentication, you're not going to get back in. So that's one of the moves from the fraudster playbook.

Now, granted, some websites don't offer alternatives, but in general, I encourage users not to choose email as a second factor unless they have MFA protection on their email account. And now an attacker with a stolen credential still needs to go ahead and compromise your multifactor authenticator, which raises the bar significantly.

Now, let's look at other ways that MFA can go bad. Imagine that you have the money and you bought YubiKeys for everybody in your organization. And you configured login to [00:14:00] require this FIDO2 compliant passwordless authentication token. Now we're at level 8. Everything's good, right? We'll always think, what could go wrong?

Other than, as they say, the one user who accidentally dropped his laptop kind of on the corner. It wasn't a big drop, enough to snap off that YubiKey and break it. It was locked out. Common attack that can be used by bad actors is to use a tool like EvilProxy, phishing as a service. which allows them to man in the middle until they steal a session cookie.

This is called doing a reverse proxy and the tackle looks something like this. And I've got the link to, resecurity. com on our notes. Number one, the user will put their password ID and password into the phishing site. Attacker grabs that, and then goes ahead and logs into the real site. At which point, the website comes back and says, Hey, you need to MFA.

And so it goes back to you and says, Hey, you need to MFA. And maybe, beep, beep, beep, beep, beep, beep, beep, out comes this message even to their phone. The user dutifully enters in their [00:15:00] MFA into the fake site, which is then goes right over to the attacker, who says, Thank you very much. And because you're still within the 30 or 60 second lifetime of that, enters in the MFA.

And they're in and, now they've got a session cookie and they're off and running. And now you can redirect the user to take them, throw them off the trail to say. Oops, server busy, or please, please try again, or something like that, that it's going to look plausible enough that you go, Okay, fine. Try it a second time.

I'm in. Good. And the user doesn't know that you've coughed up that credential and the attacker is in. Now here's an important question in a scenario like this. Do you have any observability to see that the IP address may have changed from the victim to the IP address of the bad actor? User tries to log in, they get their creds popped, bad guy comes in from some other location, and then if you do, do you instantly log out the account if you see that IP change?

Well, if you do, great, you've stopped this attack, but then you've created another problem. You see, changing [00:16:00] IP addresses is common on mobile phones. For example, there might be somebody, one of your users in New York City, who's riding the subway to and from work, or the train, or things such as that, and as you're coming into the city, you're doing that, you're going to your website on your phone, and that phone is jumping from various cell phone towers.

which means it's also going to be changing IP addresses. And if you log them out because their IP address has changed and for no other reason, then you're really providing a poor customer experience. However, if you don't log them out, then you just allowed a viable attack of stealing session cookies into your customer experience.

So pick your poison. Now we're starting to see why security can be kind of tricky. When I first heard of the scenario, I thought, well, why not look for impossible travel? For example, your user jumps from Manhattan to Moldova. Well, two things that could complicate that, that would be VPNs and IPv6.

Now, if an attacker is sophisticated and is specifically targeting a particular high value user, happens, they could use a VPN to establish a nearby point of presence that [00:17:00] looks plausible to your impossible travel rule. Hey, they went from Manhattan. Maybe the Bronx. Okay, fine, close enough. Or Manhattan to Manhattan.

There's an awful lot of points of presence you can have there. Secondly, geolocation with Microsoft for IPv6, and my observation has been notoriously bad. I see users in D. C. with their cell phone and IP address, and that IPv6 location is labeled by Microsoft and Azure as far away as Massachusetts sometimes, and they're right there.

And I don't see that with IPv4. It's pretty well understood, and I think we've got the geographic boundaries locked in, but any cell phone or its associated hotspot, more than likely, is IPv6, and it's going to land you inside this cone of confusion, and it's going to break that rule of knowing where you're at.

So, if you've found a solution to that, please let me know. All right, let's continue with the ransomware playbook. Now, once the attacker has logged into your Microsoft network, what are they going to do? You go after admin credentials, right? That's where the jackpot is. So they might find a weakness in [00:18:00] your Active Directory configuration settings, which allows them to get domain admin credentials.

And then the bad actors might go to your Azure Active Directory account, which is now called Microsoft Entra ID, and start changing some things. So, think about what a bad actor would do if they got in and they think maybe there's a clock running. First, add a Microsoft Conditional Access Policy. It says all future logins for anybody in the company, including admins, need to come from the Bad Actor's IP address range.

Now this simple conditional access policy means that everybody, except the Bad Actor, will fail at all future Active Directory logins. Oops, that's not so good. And additionally, now the Bad Actor can close all existing sessions. Therefore, every application in your organization which uses your single sign on will fail.

in the conditional access policy check. The bad actor has single sign outed your organization. And use conditional access policies to begin your nightmare. Well, we call that Single Sign Out Loss, or SSOL. So, if this happens to [00:19:00] you, you're SSOL. Now, what's worse is, the bad actor Single Sign On continues to work, so they can continue to log into things.

Now, have you thought about that exercise at your tabletop or your incident response plans?

Have you ever had a conversation with your Microsoft representative to discuss how Microsoft might respond if you ask them to override your Entra conditional policies or remove your admin accounts? Will Microsoft respond to you in a timely manner? Will they, how will they know it's you? How will they authenticate you and differentiate from an attacker who's calling in trying to socially engineer them?

What if the bad actor also deletes every YubiKey that's registered in the company? Have you thought how hard it might be to Purchase new YubiKeys, get them all re enrolled for thousands of users during a ransomware event. See, in these scenarios, bad actors are now using your greatest strengths against you.

Now, here's a little pro tip. If you're not doing this, you should do this. First of all, Microsoft recommends having no more than five global admins. Right, no matter how big your enterprise is, no more than five. [00:20:00] One of them should be an account that's used only in an emergency. It's one of these things that say, in case of fire break lasts, and then it's behind that.

So you actually have to do something. And in that particular account, you exempt it from MFA, and you put one big, long, honking password on it. You can go up to 127 characters, and you might consider using an awful lot of them. And, then take it, seal it in an envelope, sign it, lock it someplace, put it in the network room, or whatever.

So that in the event of an emergency, you've got a backdoor into your own enterprise. And that account should never be used, except for an emergency, because when it does get used, it ought to set off alarms all over the place. And so if you have thought about that, good for you. And then what I do is I go look at the sign in logs once a week.

I have a manual check. I mentioned this before Mondays over first cup of coffee or whatever. I just say, let me look at my global administrator and my senior privilege users logins just to see if everything looks good. Yeah. There's tools you can do that. Yeah. But there's nothing wrong with human eyes [00:21:00] looking at it because it doesn't take that long.

And if you would expect to say, Hey, let me look at the login activities for last month for this special account. It should always be no activity found. But if there is, or there's an attempt to log in, something's up. And oh, by the way, pick a login ID that's not going to be guessed by somebody. And so you've got a little bit of security by obscurity on that.

So that's a little pro tip that might help you out in the Microsoft environment to help protect you against that type of attack. Now, here's another MFA authentication issue that can Also get most organizations today. Let's say you have on prem servers that you use for testing your core website. And these on prem web servers are internet facing because they need to mimic a production website used by customers.

Your developers decide they want to use tools like Selenium Grid to write automated test scripts since they release daily builds. And they want to see if any new changes break things. And since test scripts can't really press a YubiKey. That's why you had the longer one, physical ones, this thing to actually activate, you have to physically contact it.

It's not a fingerprint [00:22:00] reader. It just says, Hey, I've completed a circuit. So an unattended device sitting someplace can't be used to authenticate, even though the key's in there. If you've got one of these, because a human has to. Do that. In any case, what we find then is that the scripts can't press that YubiKey.

They can't look up a pin code from Microsoft Authenticator and type it over. So what do you do? If you're a developer, just turn off MFA for the test accounts. I mean, it's a test account. Come on. This means you just need a username and a password to log in these systems since you have zero MFA for service accounts.

Now, the bad hackers can find these websites. Let's say using Shodan, because you name them WebsiteQA or WebsiteTestServer, something like that as a standard naming convention, that should definitely pick somebody's interest in going through there. If you've never played with Shodan. io, you ought to do so.

You'd be surprised what you find that's internet facing, potentially from your own organization. And I've had people who say, well, I don't want to enter my server names in there because then I'm going to reveal it to somebody. Well, you know what? It might already be revealed. It's already in their [00:23:00] database.

And so you got to decide if you want to go ahead and hope that nobody has run into it or your machine has not advertised it. Sometimes I see dual homed systems where they've got an internal and an external IP address and you didn't realize that that was left over. And now you provide different attacks.

So bad actors, if they find something like that, they might try doing what's called a password spray to get their way into your test servers. Now password spray is trying a whole bunch of passwords. You might either want to do them at a fairly slow pace. Or the most common ones so that you don't trigger any alarms or lock anything out.

But when that does occur and they get lucky or they start hitting your dev site or your QA site with every password that's, let's say, eight characters or less. You can do an exhaustive search on things that seem to be what people would type in. do you look for that on the test systems? Do you block their IP addresses after five invalid attempts?

Do you lock out the account itself so it has to do an admin reset? Probably not because it's only a test system, right? Now, are you watching the logs? For your non [00:24:00] production environment? Oh, yeah, we don't send those logs to our SIEM because it's not production and the cost of sending logs and monitoring them from non production systems, oh, it wasn't really in this year's budget.

Well, there's always the hope that these non production accounts don't have access rights to anything they shouldn't. but hope is not a good strategy in my opinion. Because otherwise, if they did, you'd fall victim to the same attack. that was used by the Russian Foreign Intelligence Service hacking group to get into Microsoft.

Now Microsoft explains what the Russian nation state actors did.Quoate: "In this observed midnight blizzard activity, the actor tailored their password spray attacks to a limited number of accounts, using a low number of attempts to evade detection and avoid account blocks based on the volume of failures. They leveraged their initial access to identify and compromise a legacy test OAuth application that had elevated access to the Microsoft corporate environment.

The actor created additional malicious OAuth applications. [00:25:00] They created a new user account to grant consent in the Microsoft corporate environment to the actor controlled malicious OAuth applications. The threat actor then used the legacy test OAuth application to grant them the Office 365 Exchange Online full access as app role, which allows access to mailboxes.

End of quote. Now in 2020, the SANS Institute suffered a data breach that exposed roughly 28, 000 PII records. And give SANS credit, they did a great job of sharing the indicators of compromise. They disclosed what they learned about the attack. They're not a publicly held company. They didn't have to, but they led by example so that others could benefit from that experience.

Now was their system overprivileged? No. Did someone grab admin creds and then dump an S3 bucket? No. So, so what happened? For those of us who remember, one of the users fell victim to what's called a consent phishing scam. Whereby the user was presented with what appears to be a SharePoint file share for an [00:26:00] Excel spreadsheet.

Happened to be labeled, copy of July bonus, xls. Well, even if you weren't expecting a bonus in July. Would be tempting to take a peek, right? Well, when the employee clicked on the open link, they were redirected to an Amazon AWS page that looked, well, you guessed it, exactly like the Microsoft login page where the user dutifully entered valid credentials.

Now, what's interesting is that allowed installation of a malicious Office 365 app that created a mail forwarding rule. And they called it Anti Spam Rule. Looks pretty innocuous to me, but at least on the label. But it actually looked for keywords such as agreement, bank, cash, dividend, payment, purchase, transfer, wire.

Kind of get the clue. And in the normal course of business, that user received emails that triggered the forwarding rule. And off went that PII. And with the Microsoft breach, what was different? They had an overprivileged test box. Once the attacker landed on that test box and enumerated all [00:27:00] permissions, there's probably a eureka moment.

Remember the Capital One breach where the attacker landed on one machine, checked for S3 keys for which it had access, and then was ultimately able to steal data directly. So, here's a thought problem for you. How do you move laterally? from an email box. I mean, in the case of Microsoft, we concluded that the test server had permissions on production 0365 as full access, and that was a serious oversight.

Why would a test system have full permissions on a production asset? Well, maybe to test sending broadcast emails to a certain group of users, but that should have been shut down very quickly, or should it turn on the day and turn off at night? Turn on the day, turn off at night. In the case of SANS, No lateral movement was required.

The attacker just sat back and enjoyed reading emails as they were forwarded by the rule that they inserted. Now, let's think about this. Microsoft did not have an MFA and legacy test system that could grant [00:28:00] access to the corporate email system. A well meaning SANS employee coughed up credentials to a legitimate looking attacker. So it seems that MFA mishaps can even get some of the smartest companies. Now, since we're playing Monday Morning Quarterback, what should Microsoft have done?

I suggest that Microsoft should have used their own solution called Managed Identities for Azure. And according to Microsoft, quote, Managed Identities provide an automatically managed identity in Microsoft Enter ID for applications to use when connecting to resources that support Microsoft Entra Authentication.

Applications can use managed identities to obtain Microsoft Entra tokens without having to manage any credentials. End of quote. Well, what does that mean? In other words, managed identities allow you to control access at the cloud level rather than at Network Layer 3, which is a whole lot better. You can check the show notes for the link with the details, but essentially the process is this.

First, you create a managed identity in [00:29:00] Azure by choosing a system assigned managed identity. or a user assigned managed identity. A system assigned managed identity creates a service principle that allows only that Azure resource to use this identity to request tokens. Nobody else can use it. A user assigned resource is a service principle that is managed separately from the resources that use it, that is to say the users, and can be used by multiple resources.

Next, you authorize that managed identity to access your target service. And some of the things you can do with these managed identities are to create, read, update, or delete, what they call CRUD, you know, typical database stuff. Use role based access control or RBAC to grant permissions and view sign in activities in Microsoft Entra sign in and logs.

So what's happening now is these users or the original accounts that's trying to log in, don't log in directly. They have to first get a managed identity, which is configured A, just for them, and then B, just for the resource that they're supposed to go to. [00:30:00] And that intermediates these access and reduces the likelihood that somebody could go ahead and hijack that token because, well, they can't really use it.

It won't work for them. Now, notice that AWS also has something similar you can do with IAM roles. If you're thinking, well, what if I can't use Azure or AWS services since I'm running on prem? Well, in this case, you should limit the IP addresses of whom can log into these machines to perhaps only Microsoft owned IP ranges.

And that way everybody else on the Internet is blocked from trying to password spray attacks. Okay, we've talked about a few MFA mishaps so far, but there's a, here's another one you might not have considered because it's not a technical attack. Your organization uses Microsoft Windows, like most companies, right?

Here's a hot new thing called Microsoft Entra Authentication, that allows you to log in with a password and a biometric login. Woohoo! You figure bad actors won't have the biometrics of your employees, so it'd be a good way to stop the bad actors. So you decide to make it mandatory across the company.

You throw the big switch because you want one consistent [00:31:00] way to log into all the devices. Now what can go wrong? Remember I said it's a non technical issue. If your company has anybody that's an Illinois state resident, they're protected by the Illinois Biometric Information Protection Act. Illinois 740 ILCS 14.

Again, in the notes. Essentially, the employee can say I do not give my consent for the company to collect my biometric data on my work phone or laptop. Now, if the company compels the employee to provide biometric data, that would be a violation of that biometric privacy law. And who knows what other jurisdictions will create similar or even more restrictive legislation in the future.

So at a minimum, you will need at least two different authentication systems for residents of Illinois. One for those who opt in to having their biometric data collected and one for those who do not grant consent. And note that that particular law also requires the organization to have a written policy that establishes the retention schedule and the guidelines for destroying biometric identifiers according to certain destruction guidelines.

And [00:32:00] you must also receive a written release executed by the subject of the biometric identifier or biometric information or the subject's legally authorized representative. If you fail in any of these items, you can expect a right of action, which your company could be sued in court for damages. Okay, well, that's a whole bunch of stuff and maybe enough for one episode, so let me just end it there.

I hope you liked learning about MFA mishaps on today's show. But remember, if used correctly, multi factor authentication is a really good safeguard to protect organizations against bad actors. However, it's also equally important, if not more important, to implement MFA correctly and then safeguard it from being turned against your organization.

Well, if you think this was an informative episode, do us a favor, share it with your friends on social media. more admins implementing MFA securely helps keep all of us safer. Also, if you'd like this type of content, don't forget to read our new newsletter. by going to cisotradecraft.substack.com [00:33:00] and you can find it there. Again, please follow us on LinkedIn because we put out a whole lot more than just podcasts and we think our information is of great value to you. We keep a high signal to low noise ratio and that is for your benefit. So thank you again for listening and thank you for increasing your cybersecurity tradecraft.

My name is G Mark Hardy and until next time, stay safe out there.