Wednesday, July 24, 2019

You don't need SMS-2FA.

I believe that SMS 2FA is wholly ineffective, and advocating for it is harmful. This post will respond to the three main arguments SMS proponents make, and propose a simpler, cheaper, more accessible and more effective solution that works today.

Just like yesterday's topic of reproducible builds, discussions about SMS-2FA get heated very quickly. I've found that SMS-2FA deployment or advocacy has been a major professional project for some people, and they take questioning it's efficacy personally.

Here are the main arguments I've heard for SMS 2FA:

- SMS 2FA can prevent phishing.
- SMS 2FA can't prevent phishing, but it can prevent "credential stuffing".
- We have data proving that SMS 2FA is effective.

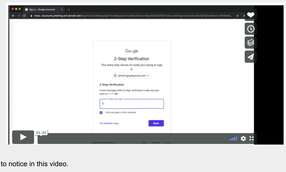
I'll cover some other weaker arguments I've heard too, but these are the important ones.

Does SMS 2FA Prevent Phishing?

I assume anyone interested in this topic already knows how phishing works, so I'll spare you the introduction. If a phishing attack successfully collects a victim's credentials, then the user must have incorrectly concluded that the site they're using is authentic.

The problem with using SMS-2FA to mitigate this problem is that there's no reason to think that after entering their credentials, they would not also enter any OTP.

I've found that lots of people find this attack difficult to visualize, even security engineers. Let's look at a demonstration video of a penetration testing tool for phishing SMS-2FA codes to see the attack in action.



There are a few key details to notice in this video.

1. The SMS received is **authentic**. It cannot be filtered, blocked or identified as part of a phishing attempt.
2. Notice the attackers console (around 1:05 in the video). For this demonstration it only contains a single session, but could store unlimited sessions. **The attacker does not have to be present during the phishing**.
3. Installing and using this software is no more complicated than installing and using a phishing kit that *doesn't* support SMS-2FA.
4. An attacker does not need to intercept or modify the SMS, in particular no "links" are added to the SMS (this is a common misconception, even from security engineers).
5. The phishing site is a pixel perfect duplicate of the original.

I think a reasonable minimum bar for any mitigation to be considered a "solution" to an attack, is that a different attack is required. As SMS-2FA can be defeated with phishing, it simply doesn't meet that bar.

To reiterate, SMS 2FA can be phished, and therefore is **not a solution to phishing**.

Does SMS 2FA Prevent "Credential Stuffing"?

Credential stuffing is when the usernames and passwords collected from one compromised site are replayed to another site. This is such a cheap and effective attack that it's a significant source of compromise.

Credential stuffing works because password reuse is astonishingly common. It's important to emphasise that if you don't reuse passwords, you are *literally immune* to credential stuffing. The argument for SMS-2FA is that credential stuffing can no longer be automated. If that were true, SMS-2FA would qualify as a solution to credential stuffing, as an attacker would need to use a new attack, such as phishing, to obtain the OTP.

Unfortunately, it doesn't work like that. When a service enables SMS-2FA, an attacker can simply move to a different service. This means that a new attack isn't necessary, just a new service. The problem is not solved or even mitigated, the user is still compromised and the problem is simply shifted around.

Doesn't the data show that SMS 2FA Works?

Vendors often report reductions in phishing and credential stuffing attacks after implementing SMS-2FA. Proponents point out that whether SMS-2FA works in theory or not is irrelevant, we can measure and see that it works in practice.

This result can be explained with simple economics.

The opportunistic attackers that use mass phishing campaigns don't care who they compromise, their goal is to extract a small amount of value from a large number of compromised accounts.

If the vendor implements SMS 2FA, the attacker is forced to upgrade their phishing tools and methodology to support SMS 2FA if they want to compromise those accounts. This is a one-off cost that might require purchasing a new phishing toolkit.

A rational phisher must now calculate if adding support for SMS 2FA will increase their victim yield enough to justify making this investment.

If only 1% of accounts enable SMS 2FA, then we can reasonably assume supporting SMS-2FA will increase victim yield by 1%. Will the revenue from a 1% higher victim yield allow the phisher to recoup their investment costs? Today, the adoption is still too low to justify that cost, and this explains why SMS 2FA enabled accounts are phished less often, it makes more sense to absorb the loss until penetration is higher.

For targeted (as opposed to opportunistic) phishing, it often does make economic sense to support SMS-2FA today, and we do see phishers implement support for SMS-2FA in their tools and processes.

Even if SMS 2FA is flawed, isn't that still "raising the bar"?

It is true that, if universally adopted, SMS 2FA would force attackers to make a one-time investment to update their tools and process.

Everyone likes the idea of irritating phishers, they're criminals who defraud and cheat innocent people. Regardless, we have to weigh the costs of creating that annoyance.

We have a finite pool of good will with which we can advocate for the implementation of new security technologies. If we spend all that good will on irritating attackers, then by the time we're ready to actually implement a solution, developers are not going to be interested.

This is the basis for my argument that SMS-2FA is not only worthless, but harmful. We're wasting what little good will we have left.

Are there better solutions than SMS 2FA?

Proponents are quick to respond that something must be done.

Here's the good news, we already have excellent solutions that actually work, are cheaper, simpler and more accessible.

If you're a security conscious user...

You don't need SMS-2FA.

You can use unique passwords, this makes you immune to credential stuffing and reduces the impact of phishing. If you use the password manager built in to modern browsers, it can effectively eliminate phishing as well.

If you use a third party password manager, you might not realize that modern browsers have password management built in with a beautiful UX. Frankly, it's harder to not use it.

Even if you can't use a password manager, it is totally acceptable to record your passwords in a paper notebook, spreadsheet, rolodex, or any other method you have available to record data. These are cheap, universally available and accessible.

This is great news: you can take matters into your own hands, with no help from anyone else you can protect yourself and your loved ones from credential stuffing.

Q. What if I install malware, can't the malware steal my password database?

Yes, but SMS-2FA (and even U2F) also don't protect against malware. For that, the best solution we have is Application Whitelisting. Therefore, this is not a good reason to use SMS-2FA.

If you're a security conscious vendor...

You don't need SMS-2FA.

You can eliminate credential stuffing attacks entirely with a cheap and effective solution.

You are currently allowing your users to choose their own password, and many of them are using the same password they use on other services. There is no other possible way your users are vulnerable to credential stuffing.

Instead, why not simply randomly generate a good password for them, and instruct them to write it down or save it in their web browser? If they lose it, they can use your existing password reset procedure.

This perfectly eliminates credential stuffing, but won't eliminate phishing (but neither will SMS-2FA).

If you also want to eliminate phishing, you have two excellent options. You can either educate your users on how to use a password manager, or deploy U2F, FIDO2, WebAuthn, etc. This can be done with hardware tokens or a smartphone.

If neither of those two options appeal to you, that doesn't mean you should deploy SMS-2FA, because **SMS-2FA doesn't work**.

Minor arguments in favor of SMS-2FA

- **SMS-2FA makes the login process slower, and that gives users more time to think about security.**

[Note: I'm not making this up, proponents really make this argument, e.g. here, here and here]

This idea is patently absurd. However, If you genuinely believe this, you don't need SMS-2FA. A simple protocol that will make login slower is to split the login process, first requesting the username and then the password.

When you receive the username, mint a signed and timestamped token and add it to a hidden form field. You can then pause before allowing the token to be submitted and requesting another token that must accompany the password.

This is far simpler than integrating SMS, as you can just modify the logic you are already using to protect against XSRF. If you are not already protecting against XSRF, my advice would be to fix that problem before implementing any dubious "slower is better" theories.

- **Attackers vary in ability, and some will not be able to upgrade their scripts.**

If you can purchase and install one kit, it is pretty reasonable to assume that you are capable of purchasing and installing another. The primary barrier here is the cost of upgrading, not hacking ability.

When adoption is high enough that it's possible to recoup those costs, phishers will certainly upgrade.

- **Don't let the perfect be the enemy of the good.**
- **Seat belts aren't perfect either, do you argue we shouldn't wear them?**
- **Etc, etc.**

This argument only works if what you're defending is good. As I've already explained, SMS-2FA is not good.

Unique Passwords and U2F are not perfect, but they are good. Unique Passwords reduce the impact of phishing, but can't eliminate it. U2F doesn't prevent malware, but does prevent phishing.

- **A phishing kit that implements SMS-2FA support is more complex than one that doesn't.**

That's true, but this complexity can be hidden from the phisher. I don't know anything about audio processing, but I can still play MP3s. I simply purchased the software and hardware from someone who does understand those topics.

- **What about "SIM swapping" attacks?**

SIM swapping attacks are a legitimate concern, but if that was the only problem with SMS-2FA, my opinion is that would not be enough to dismiss it.

- **It's not accurate to say "SMS-2FA doesn't prevent credential stuffing", because moving an attacker to other services *is* prevention.**

I have an analogy I like to use when SMS proponents make this claim. If you cover a $1cm^2$ area of your chest with sunscreen, does it prevent sunburns? I think a reasonable person would say that it does not, and you will get sunburned.

If you enable SMS-2FA, then you are still compromised, and the problem has not been prevented. Therefore, I think a reasonable neutral person would agree that SMS-2FA does not prevent credential stuffing.

Posted by taviso at 10:23 AM

No comments:

Post a Comment

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)