Multi-factor Authentication

Dolly Devashryee (dd2390), Kuan Chen (kc3422), Kai Chen (kc3443),

Travis Jon Wheelwright (tjw389), Red Mutee (rm4243)

Abstract

A privacy breach can be defined as either the disclosure of, unauthorized access to, or the misplacement of private information. There are many ways companies and organizations go about preventing a privacy breach including encrypting user passwords, storing user information in password protected databases and educating employees about the importance of cybersecurity. In order to further prevent privacy breaches from occurring, we are exploring the potential and practicality of multi-factor authentication (MFA) for all customers of a system. Multi-factor authentication is a security precaution that requires users to provide a trusted second piece of evidence in order to access their desired account. In addition, our paper goes into detail about existing MFA methods including OTP hardware tokens, OTP soft token applications, authentication through SMS, push notifications and biometrics, and gives insight on how to improve the security feature as a whole with the help of TOTP.

Keywords: OTP, MFA/2FA, TOTP

Introduction

Privacy breaches are the mass exposure of confidential data to an untrustworthy third party or environment. In other terms, privacy breaches occur when personal data, which may or may not have been private, is made or is discovered to be public. Such breaches are usually a consequence of vulnerable implementations of a system, vulnerable operations of a system, or vulnerable business procedures involving a system. Although there are many ways companies try to beat these vulnerabilities, mass privacy breaches are still very common with amounts increasing yearly jeopardizing sensitive information of millions [6]. This paper is primarily motivated by the violations caused by privacy breaches, in hopes to lower annual numbers of breaches and to overall improve security.

In addition to security measures taken today, our paper explores multi-factor authentication (MFA) for when users want to access their account as a way to further heighten user security. MFA requires that users provide a second piece of known information that will verify the user's true identity. With MFA, there is a second layer of protection because users are unable to access an account with just a password and need to communicate from a second, trusted device. We specifically propose the use of a certain type of MFA called Time-based One-time Password (TOTP) which generates a random passcode based on the current time that a user needs to give before they can access an account. This is especially safe because the TOTP passcode changes with time every 30 to 60 seconds [11].

Background

Privacy breaches can be caused by a number of problems: whether it be weak passwords, default passwords, no passwords required, etc. If a threat actor is able to gain access to data that he or she is not authorized to access, a privacy breach has occurred. Privacy breaches happen when there is a gap or weak spot in the security of a company/product/device.

There are a variety of solutions out there but not all are "created equal". Changing default passwords is a start. You would think that in the year 2018, this would go without saying. A week after the huge Equifax hack in

2017, affecting millions of Americans and Canadians, it was discovered that Equifax Argentina had suffered a privacy breach as well. "…an online portal designed to let Equifax employees in Argentina manage credit report disputes from consumers in that country was wide open, protected by perhaps the most easy-to-guess password combination ever: "admin/admin." [7]

There is a better solution than just changing default credentials: having longer and more complex password requirements. Passwords are either stolen via keyloggers / phishing attacks or are brute forced. (This also brings up the solution of limiting login attempts and locking out accounts, but that is not the focus of this paper.) The longer the password or passphrase is, the longer it is going to take to brute force it. Phishing / keylogging requires employee training.

The best solution of them all is to use two-factor authentication. Wikipedia has this to say of two-factor authentication: "It is a method of confirming users' claimed identities by using a combination of two different factors: 1) something they know, 2) something they have, or 3) something they are." [8] Something you know would be like a standard password. Something you have would be like a phone with an authenticator app or a key fob. Something you are could be biometric data, for example.

Not only does it require a password, but it also requires a second proof that you are who you say you are. Implementing this technology (almost) eliminates phishing / keylogging / brute force attempts to steal your login credentials. Before smartphone apps like Google Authenticator or Duo Mobile, people used phone calls or SMS text messages to verify their identity by receiving a one-time passcode to use to log in. Their "something you have", if you will. This is still an option offered today, but is not recommended as it is not entirely secure.

SMS text messages are not sent nor stored in an encrypted form. They are susceptible to eavesdropping. Granted, a device such a StingRay is not commonly found in a hacker's tool kit. They are extremely expensive, but nevertheless they are still available, so they cannot be ruled out. I don't believe, however, that a text or a phone call even constitutes being a second factor of authentication. A code sent to your phone is more like *two-step* authentication, not *two-factor* authentication. It has been easily proven that an attacker can socially engineer an employee of one's cell phone provider, and request that a phone number be transferred to a device or SIM card that does not belong to the original owner [9]. All texts and phone calls, (even those containing these authentication codes), are now being intercepted and routed to the phone of an attacker. Having a phone number as an authentication method is a poor choice, and it certainly does not prove an identity.

Taking into consideration all of the previously mentioned solutions, I believe that the best method that is publicly available is to use is either authentication applications or physical 2FA tokens, such as a Yubikey. These are truly "something you have". Only if the attacker has physical access to these devices could they compromise login credentials. Even if an attacker achieves physical access, one can easily dissociate the app with the account or with the token, making the app or device useless. Recently an article was published about the positive benefits of using these security tokens. Google has issued Yubikeys to each one of their employees. Their results were quite positive. They had this to say: "None of Google's 85,000 employees have had their work accounts compromised since it started requiring security keys to log in". [10] The numbers speak for themselves.

Threat model

In the previous report, we propose to use MFA to protect user information and privacy. Although there are many ways to implement MFA, and the safety effects are naturally quite different. Below we analyze common MFA methods to discuss their threats.

1. OTP hardware token

For this authentication method, the user possesses a hardware token in addition to knowledge of their account's username and password. Those secret values

can be stolen from the server or from the hardware token vendor. Token provider RSA, for example, was breached with millions of token serial numbers stolen, exposing customers to hack attacks.

2. OTP soft token applications

For this authentication method, a user receives an OTP from a pre-installed application. The user then inserts this password into the authentication form. For OTP soft token applications, its advantages and disadvantages are similar to OTP hardware token.

3. OTP via SMS

For this authentication method, the user receives an SMS text message with a random code (time limited OTP). Then the user needs to enter the code into the same authentication portal or application in which he or she has entered the user name and password. However, this method is the most vulnerable with security flaws and a less than desirable user experience.For SMS, there are two weakness. First, hackers can hijack SMS messages in transit. Another weakness is that the expecting user is open to receive SMS from anyone – and fraudulent authentication messages are a popular tactic in phishing attacks.[1]

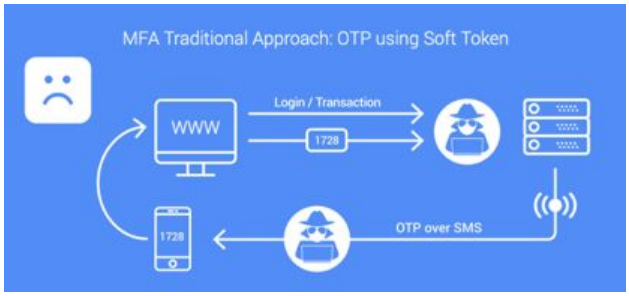

Figure 1. Threats of SMS and Token

In addition, it is not convenient to use SMS.When using SMS OTPs, users must copy the OTP information from the receiving device to the login form.

4. Push notification

The problem with push notification is that the encryption is done in two stages: service provider to push server and push server to mobile device. The plain data is revealed at the push server, which means confidential data, including OTP, cannot be sent over push notification. Push notification should only be used to wake up the application so it can send an inquiry to the server. In which case, the authentication would use PKI, but this in turn is vulnerable to SSL weaknesses.[1]

5. Biometrics

Static biometric is also considered relatively inaccurate.Fingerprints, for example, can now be copied.

Based on our analysis to each MFA method, We could get the the Attack Tree Model like figure 2. And we will discuss how to design a better MFA in the next part.

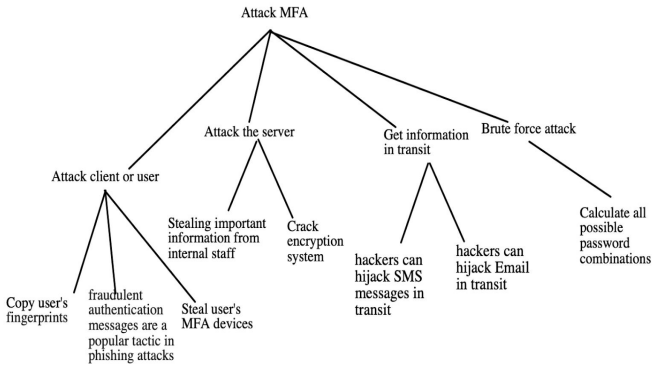

Figure 2. Attack tree for MFA

Design

Passwords are the most common method of authentication, but they are not secure and are easy to leak and impersonate. More and more places require two-factor authentication (2FA) or multi-factor authentication(MFA).

In general, three different types of evidence can prove a person's identity[2].

- *Secret information*: Certain information that only user knows, such as a password.
- *Personal items*: The user's personal items, such as ID cards, keys.
- *Physiological characteristics*: the genetic characteristics of the user, such as fingerprints, appearance, etc.

These evidences are called three "factors". The more factors there are, the stronger the proof and the more reliable the identity.

Two-factor authentication refers to the evidence that two factors are required to pass the certification. Bank cards are the most common two-factor authentication. The user must provide both a bank card and a password in order to withdraw cash.

A common two-factor combination is a password + a personal item, such as the USB key of an online bank. Users can plug in the USB and enter the password to log in to the online bank. However, users are not allowed to carry the USB at all times, and the mobile phone is the best alternative. Password + mobile phone is the best two-factor authentication solution.

However, mobile phone messages are not secure, because the messages can be easily intercepted and forged, and even SIM cards can be cloned[3]. There have been cases where the identity card was forged first, and then the same mobile phone number was applied to transfer the money away[4].

Therefore, a secure two-factor authentication is not a password + a short mobile phone message, but a TOTP as described below.

The Time-based One-Time Password algorithm (TOTP) is an extension of the HMAC-based One-time Password algorithm (HOTP) and it generates a one-time password by instead taking uniqueness from the current time[5]. It has been adopted as Internet Engineering Task Force standard RFC 6238, is the cornerstone of Initiative For Open Authentication (OATH), and is used in a number of two-factor authentication systems.
Its steps are as follow:

1. User turns on two-factor authentication, the server generates a key.
2. The server prompts the user to scan the QR code (or use other methods) to save the key to the user's mobile phone. In other words, the server and the user's mobile phone now have the same key. The key must be bound to the phone. Once the user changes the phone, a new key must be generated.
3. When the user logs in, the mobile client uses the key and the current timestamp to generate a hash value, whose valid period is 30 seconds by default. The user submits this hash to the server during the valid period.
4. Server also uses the key and the current timestamp to generate a hash that matches the hash submitted by the user. As long as the two do not match, server refuses this user to log in.

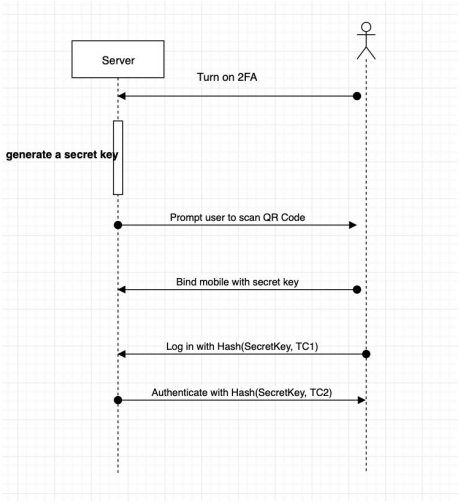

Figure 3. Illustration of TOTP

Implementation

It's important for TOTP algorithm to make mobile clients and servers to get the same hash during 30 seconds period.

$$TC = floor((unixtime(T_{now}) - unixtime(T_0))/TS)$$

In the above formula, TC represents a time counter, $unixtime(T_{now})$ is the current Unix timestamp, and $unixtime(T_0)$ is the timestamp of the agreed start time. The default is 0, which is January 1, 1970. TS is the length of time for the hash to be valid. The default is 30 seconds. Therefore, the above formula becomes the following form.

$$TC = floor(unixtime(T_{now})/30)$$

Therefore, as long as it is within 30 seconds, the value of

TC is the same. The premise is that the server and mobile phone time must be synchronized.
Next, calculate the hash.

$$TOTP = HASH(SecretKey,\ TC)$$

In the above code, HASH is the agreed hash function, the default is SHA-1.

```
var tfa = require('2fa');

tfa.generateKey(32, function(err, key) {
    console.log(key); // awba6da9dzvorfwip9wxqghs6mqcwda0
    var tc = Math.floor(Date.now / 1000 / 30);
    var totp = tfa.generateCode(key, tc);
    console.log(totp); // 059384
});
```

Figure 4. TOTP In Javascript

Evaluation

With free software available online for your PC and on the appstore for your mobile device, TOTP is a fast, cost-effective and efficient way to implement two-step verification (2FA). Software such as DUO Mobile are widely used in universities, large corporations and by individuals wishing to have secure authentication for multiple platforms. To determine the reliability of any software, potential weaknesses and vulnerabilities need to be explored and examined.

- Just as passwords can be phished, TOTP values can also be phished by attackers. However, the attackers must proxy the credentials in real-time as the TOTP values are only valid for very small amounts of time. As a result, it is much harder to phish TOTP values as attackers can not collect credentials without worrying about time restrictions. In other words, if an attacker was able to intercept both the access code and the user's passwords, they would have a short window of time where they could use the information.

- The shared secret or secret key can also be stolen from the user's device if the user's device is breached. For example, having a mobile device with malware that identifies the users secret key would invalidate TOTP's main security design principle: the secret key has to be secret!

- TOTP relies on UNIX TimeStamps and although it is unlikely, it is possible for the clocks of the user's device and the server's to de-sync. As a result the server will typically accept one-time passwords generated from timestamps that differ by ±1 time interval from the client's timestamp.

- Implementations of TOTP must also have a limit on login attempts orelse the system will be vulnerable to brute-forcing of values. It is also important to note that TOTP, like all other OTP, does not prevent session hijacking.

Related Solutions

The two main standards for One-Time Password are HOTP and TOTP. In both HOTP and TOTP, a token is used to generate a 6-digit or 8-digit numeric code. HOTP (HMAC-OTP), also known as Event-based OTP, relies on two pieces of information like TOTP. The first piece is the secret key and it's only know by the server that validates submitted OTP codes and by the token. The second piece is the moving factor. The moving factor for HOTP is a counter and the moving factor for TOTP is time. To calculate an OTP, the token feeds the moving factor into the HMAC algorithm using the token seed as the key. This produces a 160-bit values that is then converted to a 6 or 8 digit numerical value.

The main difference between HOTP and OTP is that while the OTM schemes generate single-use codes, in HOTP, a generated OTP is valid until it is used or until a subsequent OTP is used. In HOTP there are a number of valid "next OTP" codes. This is because the button on the token can be pressed, thus incrementing the counter on the token, without the resulting OTP being submitted to the validating server. For this reason, HOTP validating servers accept a range of OTPs known as the

validation window. OTP that are generated by a counter that is within a set number of increments from the previous counter value stored on the server will be accepted by the server. If the token counter is not in the validation window, the validation fails and the token must be re-synchronized. Unlike HOTP, in TOTP, there is only one valid OTP generated form the current UNIX time at any given time and it expires within a set amount of time. From a security perspective, TOTP seems to be the better the choice.

Conclusion

In this writeup, we discussed solutions to the very hot-button issue of privacy breaches. It seems that every week or so, a major corporation is hacked and all of their customer's data are mass spread around the internet. Our proposed solution of two-factor authentication will solve most of the causes for security breaches, (along with good basic security practices). In place of a single authentication factor, i.e a password, users must present a secondary method to prove that they are who they say they are. In reality, this would be very difficult for a threat agent to get their hands on. Since a user will only be authenticated if both factors are present, this will stop the majority of attacks from occurring. As we've outlined in this paper, not all two-factor solutions are as secure as others. The Time-based One-Time Password algorithm (TOTP) offers the security needed with a specific time-based key, and time-limited window for authentication. We believe this to be the best solution to the problem.

References

[1] MFA-Method

https://doubleoctopus.com/wp-content/uploads/2017/05/How-to-Choose-MFA-Method.pdf

[2] Three types of factor auth

https://www.globalknowledge.com/blog/2018/06/26/multi-factor-authentication/

[3] How to clone SIM card

https://www.tech2hack.com/how-to-clone-sim-card-easily/

[4] Steal phone number

https://www.howtogeek.com/358352/criminals-can-steal-your-phone-number.-heres-how-to-stop-them/

[5] An intro to TOTP

https://en.wikipedia.org/wiki/Time-based_One-time_Password_algorithm

[6] Data Breach Statistic

https://www.nbcnews.com/business/consumer/data-breaches-happening-record-pace-report-finds-n785881

[7] Equifax Argentina Default Credentials

https://krebsonsecurity.com/2017/09/ayuda-help-equifax-has-my-data/

[8] Basic Multi-Factor Definition

https://en.wikipedia.org/wiki/Multi-factor_authentication

[9] Insecurities of Relying on Phone Numbers

https://www.inc.com/joseph-steinberg/do-not-let-criminals-steal-your-phone-without-having-to-actually-take-the-device.html

[10] Google's Use Of YubiKeys
https://www.businessinsider.com/none-of-googles-employees-get-phished-because-of-yubikey-security-key-2018-7

[11] What is TOTP?

https://searchsecurity.techtarget.com/definition/time-based-one-time-password-TOTP