CS 4404 - Mission 1

Matthew Hagan, Peter Maida, Tim Winters

November 2019

1 Introduction

Two-factor authentication (2FA) can be used to provide an added layer of security to a system that requires authentication. While most authentication systems require a password or pin, 2FA is becoming increasingly popular as a countermeasure to weak passwords and information leaks. The second factor acts as another method of verifying authenticity. In this paper, we will talk about SMS, biometrics, and geolocations as options for 2FA. We then describe and implement the SMS 2FA authentication system on simulated virtual machines. Once the infrastructure is created, we break the SMS 2FA by a Denial of Service (DoS) with a middlebox. To defend against this attack, we improved our infrastructure by adding the security goals of confidentiality and authenticity. The new model includes encrypted and verified authentication codes to surpass our previous attack vector.

2 Reconnaissance

2.1 SMS Authentication

Short Messaging Service (SMS) is a mobile communication protocol used for sending short text messages between mobile devices among other things. The protocol was first defined in 1985, with the first message being sent in 1992 [1].

SMS authentication is popular because cellphone numbers are unique to a single person, so authentication is easy to verify. Unfortunately, this puts a lot of trust in SMS-based auth, and if an attack vector is found, it can be easy to impersonate another user's authenticity. This could then lead to attacks to other platforms that rely on a user verifying their identity via SMS such as bank accounts, social media, and email.

2.1.1 SS7

Signaling System 7 (Common Channel Signalling System 7 in the US) is the set of protocols that define communications between cell phone networks when making phone calls or sending messages to one another [2].

Because the security is low for SS7, once they have access to the SS7 system, a hacker can essentially have access to the same amount of information and snooping capabilities as security services. [3] They have the potential to forward your call to a recording station, read messages sent between phones and track your location [3].

2.1.2 SMS vs IP

The popular Apple Inc. iMessage protocol (and other web-based messaging platforms) is similar to SMS, in that it allows users to transmit short text messages between mobile devices. The difference between SMS and iMessage and others is that iMessage is internet based, meaning it does not transmit over phone lines like SMS. Instead, it transmits over data lines. iMessage and other internet based communications are encrypted [4] and will not be the targets of this attack.

2.2 Authentication Workflow

SMS authentication works under the principle that the only access to the SMS is by the owner of the

phone. If a user's phone was thought to be compromised, then an SMS text would not be authentic.

Alice's login process using SMS as a second factor would look like this

1. Alice logs in to a website using a password
2. If the website feels the need to verify her identity via a second method, it would send an SMS to her phone
3. Alice receives an unencrypted message containing a code sent by the website
4. Alice provides the received code to the website
5. If the codes are the same, the website assumes that Alice is the one logging
6. Alice is allowed to log in

We will be exploiting the third step of this process by intercepting the SMS key and acting as if we are Alice. Because the SMS is not encrypted, we can read the plaintext, change it, and then send the malformed code to Alice preventing her from being able to log in. This is an example of a Denial of Service attack.

2.3 Biometric Authentication

Biometric authentication is a security factor that relies on the premise of "what you are". It typically is done through primitive methods such as fingerprint recognition or facial recognition but can oscillate to more complicated methods such as requiring DNA for authentication [5].

Some of the foremost mentioned methods of biometric authentication are popular because of their lack of additional effort to authenticate. For fingerprint recognition, the user simply needs to place their finger on a sensor conveniently located on the device [6]. For facial recognition, the user only needs to look at the device to have it authenticate [7]. The most complicated part of this method for the user is the initial setup, as it requires a few minutes for the user to step through the automated calibration steps [7].

For the remainder of this section, we will be focusing on Apple mobile device biometric authentication, as they hold 47% of the mobile device market share [8] and thus are the most well-known use case to investigate.

2.3.1 Fingerprint Recognition

In 2012, Apple was looking to streamline the iPhone login process. As such, they acquired AuthenTec [9] and their engineers to make what is now known as Touch ID [6].

While Touch ID appeared to be more secure to the average user, to the trained professional it was a vulnerability waiting to be exploited. From a mathematical standpoint, Apple said that Touch ID has a 1 in 50,000 chance of an accidental login occurring [10]. That means that for every 50,001 people in a population, one other person will likely be able to unlock your device by just applying their fingerprint. This attack vector would be considered an attack via enumeration (similar to User enumeration or bruteforcing) [11]. This is a less likely attack vector, as it requires collecting a large set of the population (or at least their fingerprints). On the other hand, using a simple Google query of "touch ID Hacks" will list a plethora of different attacks and bypass methods that exist ranging from ArsTechnica [12] to the various other studies [13] [14]. Despite its given flaws, it is still workable for the average user, given that reputable device manufacturers like Samsung have included it in their most recent flagship device [15].

2.3.2 Facial Recognition

As fingerprint recognition has proved to be a controversial 2FA security method on mobile devices, manufactures such as Apple have moved towards more rigorous and exciting methods such as facial recognition. Before Apple's Face ID, other device manufacturers performed facial recognition by doing image comparison/recognition of a cached image and the current feed of the video camera [16]. Modern device manufacturers have shifted to more advanced methods of

authenticating facial recognition, as the image comparison method of authentication was proved to be easily spoofed [17]. Apple's Face ID, being one of the more advanced methods of biometric authentication is not without its flaws. Before highlighting those, we would like to enumerate on the positive attributes of Apple's Face ID, as it is currently one of the stronger methods of mobile biometric authentication. Face ID works through a series of cameras. To start, the normal user facing camera looks for the individual's face and if it is able to spot a face, an embedded dot projector projects a series of dots onto the individual's face. The embedded IR camera then uses the depth of the dot map to compare to previous cached models. If the cached model aligns well enough and the individual is looking at the camera, the device will deem the user authentic and unlock. This can all be found in Apple's online support documention on Face ID Technology [18].

Even though this mobile security method is ahead of other device facial recognition methods on the market [19], it isn't without its flaws. Soon after Face ID was initially released, it was able to be broken into by using a 3D printed mask [20]. While extreme, this exploit proved again that the "What you are" aspect of mobile security is yet again able to be exploited. More recently, an even more glaring security flaw was found relating to how Face ID works with glasses. Security professionals were able to authenticate into the iPhone by applying black tape and a white dot to a pair of glasses, and then wearing that pair [21].

For Face ID, there are various controls and countermeasures that can be put in place. Before the more recent (and glaring) black tape security flaw was found [21], if somebody wanted to get into a Face ID encrypted phone, they either need to have the passcode, the cached face, or own advanced methods of scanning and printing the desired users face.

2.4 Geolocation Authentication

Another factor of authentication is location. Location for a specific user is highly sensitive data, but efficient when used for authentication [22]. The location information can be IP addresses or latitude and longitude. With Global Position System (GPS) information, a server can detect whether the computer and phone are located in the same environment, but geolocation information is not always accurate. For example, when the device is located in a VPN network or an enterprise's large management network, the geolocation is not the real location of the device, which would result in an authentication failure [23].

A common use for location information with some companies, such as Google, is for when an account sign in is detected from a foreign location or device. The user will be sent an email confirmation to verify that this location is valid and you are the one signing into the account.

2.4.1 Geolocation Vulnerability

Geolocation data isn't an inherently secure means to provide authentication because coordinates can be guessed if you know details about the identity you are trying to impersonate.

Another way geolocation authentication can be broken is when the attacker logs into your account and the 2FA device being located is within a valid geofence. This is a confused-deputy problem because while you aren't logging in, your device is still in the correct location and thus allows the criminal access without your consent. To solve this problem, additional security is required to ensure that the device you use to log has the security to detect foreign IP addresses contacting it [24].

2.4.2 Geolocation Ease of Use

One form of location based authentication uses the location of a secondary mobile device to authenticate if the device is in close proximity to the client requesting login access [25].

Another example of having a device in close proximity to the client is with an ambient noise sent between both microphones. Karapanos et al. create a system called Sound-Proof that uses the proximity between the users mobile phone and computer without the need for user-phone interaction. The

proximity is compared using ambient noises recorded by their microphones. The sounds are silent to the user, so it portrays itself as a one-factor authentication method. This method is easily deployed and a validate with empirical evidence that ambient noise is a robust discriminant to determine the proximity of two devices [26].

3 Infrastructure

3.1 Setting up the VMs

This section describes how to set up the server (VM 1) the client (VM 2), and the meddler in the middle (VM 3) All of the commands in this section should be run as root

3.1.1 VM 1

This VM will contain the 'server' that sends the authentication code to the client. We will need to give it an IP address that is outside of the subnet using the following command. Then, to prevent ICMP requests from getting through, we need to drop them, and finally add our man-in-the-middle as a route so that packets don't go directly to our destination.
See the following commands:

```
$ ifconfig eth0:0 10.4.X.65/26 up
$ sysctl net.ipv4.conf.all.accept_redirects
    ↪ =0
$ /etc/init.d/networking restart
$ for f in /proc/sys/net/ipv4/conf/*/
    ↪ accept_redirects; do echo 0> $f;
    ↪ done
$ ip route add 10.4.X.128/26 via 10.4.X.66
$ ip route add 10.4.X.129 via 10.4.X.66
```

3.1.2 VM 2

This VM will contain the 'client'. The setup is very similar to VM 1
See the following commands:

```
$ ifconfig eth0:0 10.4.X.129/26 up
$ sysctl net.ipv4.conf.all.accept_redirects
    ↪ =0
$ /etc/init.d/networking restart
$ for f in /proc/sys/net/ipv4/conf/*/
    ↪ accept_redirects; do echo 0 > $f;
    ↪ done
$ ip route add 10.4.X.64/26 via 10.4.X.130
$ ip route add 10.4.X.65 via 10.4.X.130
```

For VM 1 and VM 2 you will need to install the pycrypto library from PyPI using python3, as both of these scripts run with Python3. This may require the installation of `build-essential` and `python3-dev`.

3.1.3 VM 3

This VM will contain the 'man-in-the-middle' that intercepts communication between the client and server.

The setup is similar to the other two machines, except we will be giving this machine two aliases so that it can communicate on both subnets, as well as enabling IP forwarding. Finally, we add a route to intercept forwarded packets and send them to the NetFilterQueue script.
See the following commands:

```
$ ifconfig eth0:0 10.4.X.66/26 up
$ ifconfig eth0:1 10.4.X.130/26 up
$ sysctl net.ipv4.ip_forward=1
$ /etc/init.d/networking restart
$ iptables -I FORWARD -j NFQUEUE --queue-
    ↪ num 1
```

For this VM, you will also need to install the NetfilterQueue and Scapy packages from PyPI. Follow the instructions on the NetfilterQueue PyPI page. https://pypi.org/project/NetfilterQueue/

Scapy requires `setuptools`. To install, `apt install python-pip`

3.1.4 Flushing Arp Cache

Once you have set up each VM, flush the ARP cache on each VM to clear current routing data.

```
$ ip -s -s neigh flush all
```

3.2 Setting up our architecture

The scripts that we used can all be found on Github. There are two versions of both the client and server scripts, an insecure version, and a secure version. The insecure version replicates SMS messages sent in the real world with the data sent from server to client in plain text. The secure version uses RSA key signing to provide the goal of data authenticity. In addition, the secure version provides RSA public key encryption of the message code to provide the goal of data confidentiality.

The network intercepter script should be run with Python2 on VM 3. This script will parse the data in the packet and modify it if the packet contains the text `Auth Code:` .

To run the attack, start the server first, then start the network interceptor on the middleman machine, and finally run the client. Received bytes will be displayed on the client, and the original message will be displayed in the

4 Attack

4.1 Attack Vector

For this attack will will be assuming that the infiltrator has gotten access to the SS7 network, and is able to view SMS messages being sent from Sender A, as well as send messages as Sender A to Recipient B. With this access, we can perform a denial of service attack by changing the bytes sent. When Recipient B receives the authentication code, it will be incorrect, and he will not be authorized. Thus a Denial of Service (DoS) attack has been successfully performed. By sending an incorrect code that looks indistinguishable from a correct code, the recipient will not be able to detect that there was interference.

4.2 Carrying out the Attack

4.2.1 Interception of packet

We used the NetFilterQueue library to perform our attack. This library allows us to read the packet data, modify it if needed, and choose to either 'accept' the packet or 'drop' the packet. For this attack, we want the attacker to remain undetected, so we will not drop any packets.

4.2.2 Modification of Payload

In order to deny service to the client, the attacker will modify the payload so that an incorrect payload is received. There is no checksum or digital signature attached, so the recipient has no knowledge any modification took place.

For this attack, the code is simply reversed. If the server sends `Auth Code: 123456` the client will receive `Auth Code: 654321`. Because 2FA codes are generally 6 digits in a random order, reversing the bits looks just as random. Different algorithms to generate a different code could be used with the same infrastructure if different outcomes are desired.

4.3 Why it's Worth Defending

By not addressing this vulnerability, a user can easily be locked out of their account if they use SMS for two factor authentication. A malicious actor can detect the plain text authorization code being sent from a server to a client and change it to be something invalid. The user would never be able to pass the second factor of authentication, and even worse, would not know why because the code looks valid.

5 Defense

To defend against the attack, we are implementing two forms of defense:

1. Encrypt messages using public-key infrastructure. This prevents an attacker from easily determining what bytes to change to send a seemingly authentic message. Now, changing a single

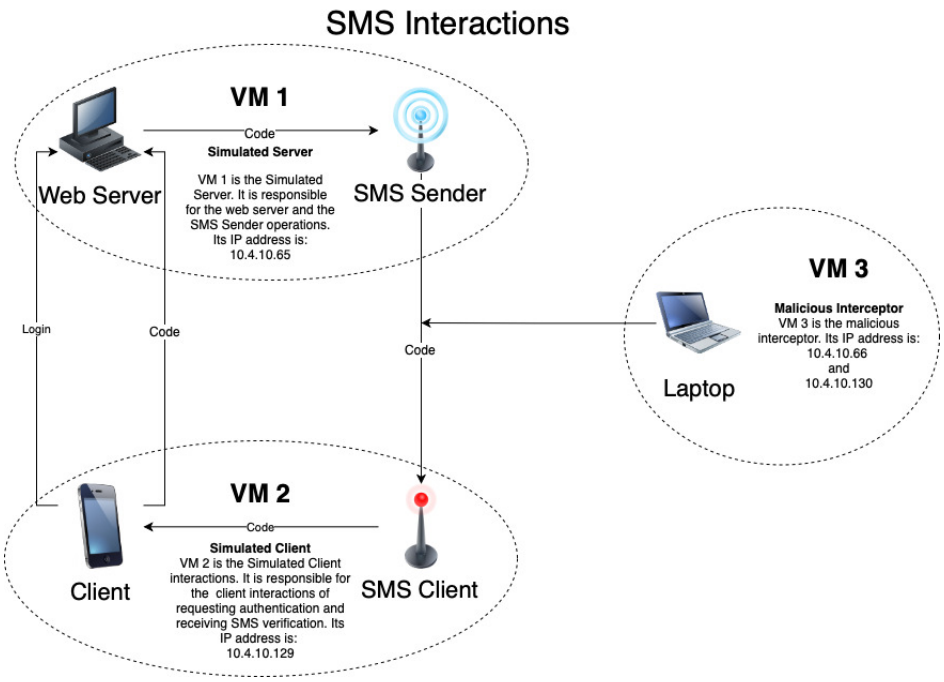

Figure 1: The model showcasing the route of the 2FA code.

bit will change the entire output of the message, and it will be easy to detect that someone is interfering.

2. Digitally sign the encrypted message. By adding a digital signature via PKI, the recipient can easily detect if the message can be determined authentic or not. This will prevent the attacker from sending their own encrypted message with an incorrect code.

For both the encryption and digital signing we implemented public-key encryption via RSA. Public key encryption means we don't need to insecurely transmit a secret key. In practice, the public key can be stored in a database with the cell network, and the private key can be securely stored in the SIM card.

You can see the encryption in action in the `sms_*_secure.py` scripts. The server encrypted and signed messages sent to the client, and the client verifies the signature and prints to the console that the message is a authentic. At the meddler-in-the-middle, his script is unable to detect that an authorization code was sent, and therefore is unable to properly meddle with the SMS two factor authentication.

6 Conclusion

This attack demonstrates how a third party that has access to the SS7 network could execute a denial of service attack on a user trying to authenticate using SMS. In preparing the attack, we learned about IP forwarding and alias, which we didn't know could be done, as well as how to use an existing tool (netfilterqueue) to intercept packets, modify them, and send them again. After learning about public-key encryption, we decided that it would be the best way to defend against an attack like this because phones have unique SIM cards that could act as the private key in a public/private pair, and the cell network could act as a CA for each customer's public key. Breaking RSA encryption is outside the scope of this course, but we feel that it is a good protective measure against SMS snooping.

References

[1] "Hppy bthdy txt!" Dec 3, 2002. [Online]. Available: http://news.bbc.co.uk/2/hi/uk_news/2538083.stm

[2] P. E. Company, "Into to ss7 signaling¡br¿," Tech. Rep. [Online]. Available: https://www.patton.com/whitepapers/Intro_to_SS7_Tutorial.pdf

[3] S. Gibbs, "Ss7 hack explained: what can you do about it?" Apr 19, 2016. [Online]. Available: https://www.theguardian.com/technology/2016/apr/19/ss7-hack-explained-mobile-phone-vulnerability-snooping-texts-calls

[4] "New version of ios includes notification center, imessage, newsstand, twitter integration among 200 new features," Jun 6, 2011. [Online]. Available: https://www.apple.com/newsroom/2011/06/06New-Version-of-iOS-Includes-Notification-Center-iMessage-Newsstand-Twitter-Integration-Among-200-New-Feature

[5] "New software can verify someone's identity by their dna in minutes." [Online]. Available: https://phys.org/news/2017-11-software-identity-dna-minutes.html

[6] "Use touch id on iphone and ipad." [Online]. Available: https://support.apple.com/en-us/HT201371

[7] "About face id advanced technology." [Online]. Available: https://support.apple.com/en-us/HT208108

[8] O. Sohail, "Apple's market share in the premium smartphone segment remains unrivaled, despite a drop in sales," -06-19T15:30:13+00:00 2019. [Online]. Available: https://wccftech.com/apple-market-share-premium-smartphone-segment-q1-2019/

[9] E. M. Rusli, "Apple to acquire authentec for $356 million," 1343395000 1343395000. [Online]. Available: https://dealbook.nytimes.com/2012/07/27/apple-to-acquire-authentec-for-356-million/

[10] H. Guinness, "How secure are face id and touch id?" [Online]. Available: https://www.howtogeek.com/350676/how-secure-are-face-id-and-touch-id/

[11] "User enumeration explained: Techniques and prevention tips," -06-15T14:04:58.000Z 2017. [Online]. Available: https://blog.rapid7.com/2017/06/15/about-user-enumeration/

[12] D. Goodin, "Bypassing touchid was "no challenge at all," hacker tells ars," 9/24/ 2013. [Online]. Available: https://arstechnica.com/information-technology/2013/09/touchid-hack-was-no-challenge-at-all-hacker-tells-ars/

[13] "Apple iphone 6 touch id hacked." [Online]. Available: https://www.bankinfosecurity.com/apple-iphone-6-touch-id-hacked-a-7348

[14] R. Brandom, "Your phone's biggest vulnerability is your fingerprint," -05-02T08:00:02-04:00 2016. [Online]. Available: https://www.theverge.com/2016/5/2/11540962/iphone-samsung-fingerprint-duplicate-hack-security

[15] "What is the ultrasonic fingerprint scanner on galaxy s10?" [Online]. Available: https://www.samsung.com/global/galaxy/what-is/ultrasonic-fingerprint/

[16] "How does face recognition work on galaxy s10e, s10, s10+ and galaxy fold?" [Online]. Available: https://www.samsung.com/global/galaxy/what-is/face-recognition/

[17] R. Amadeo, "Galaxy s8 face recognition already defeated with a simple picture," 3/31/ 2017. [Online]. Available: https://arstechnica.com/gadgets/2017/03/video-shows-galaxy-s8-face-recognition-can-be-defeated-with-a-picture/

[18] "About face id advanced technology." [Online]. Available: https://support.apple.com/en-us/HT208108

[19] G. Gottsegen, "iphone x's face id is years ahead of android competition." [Online]. Available: https://www.cnet.com/news/iphone-x-face-id-2-years-ahead-of-android-report/

[20] S. M. i. S. o. N. 15 and . A. Pst, "iphone's face id can be hacked, but here's why nobody needs to panic." [Online]. Available: https://www.techrepublic.com/article/iphones-face-id-can-be-hacked-but-heres-why-nobody-needs-to-panic/

[21] D. Winder, "Apple's iphone faceid hacked in less than 120 seconds." [Online]. Available: https://www.forbes.com/sites/daveywinder/2019/08/10/apples-iphone-faceid-hacked-in-less-than-120-seconds/

[22] A. R. C. H. C. P. Shraddha D. Ghogare, Swati P. Jadhav, "Location based authentication: A new approach towards providing security." [Online]. Available: https://pdfs.semanticscholar.org/1c9a/c1ead334ee6320f2ab26ee7afadfa6cafa07.pdf

[23] J. Zhang, X. Tan, X. Wang, A. Yan, and Z. Qin, "T2fa: Transparent two-factor authentication," *IEEE Access*, vol. 6, pp. 32 677–32 686, 2018, iD: ieee$_s$8386653.

[24] M. Stanislav, "Two-factor authentication." [Online]. Available: http://books.google.com/books?id=3EU3DwAAQBAJ

[25] B. Trinh, C. Ray, and C. Srichankrad, "Geolocation-based two-factor authentication," 2019-02-12. [Online]. Available: http://www.freepatentsonline.com/10206099.html

[26] N. Karapanos, C. Marforio, C. Soriente, and S. Capkun, "Sound-proof: Usable two-factor authentication based on ambient sound," *arXiv.org*, 2015. [Online]. Available: https://www.usenix.org/system/files/conference/usenixsecurity15/sec15-paper-karapanos.pdf