Broken Promises? An Analysis of Mobile Apps Promising User Privacy

December 10, 2013
By Andrew Jenkins

Mobile Apps Promising Privacy

Abstract: A recent trend in the mobile marketplace is an influx of apps that promise additional measures of privacy and anonymity to users. Most prominent in this space is Snapchat, which allows users to send temporary pictures to users' contacts and discourages the use of screen captures. Although the app intends to ensure the confidentiality of these images, much analysis has shown numerous holes in its security model. Perhaps more alarming are anonymous social networks like Whisper, which allow user to share content and images with other users completely anonymously. Not only are these apps worrisome for parents and authorities who have little control over what people share with such services, but also for the users who have no idea how safe their data really is. These concerns will be addressed in this paper.

Introduction

Recently, many consumer-oriented software applications have advertised features that increase user privacy or anonymity. This trend has been especially prevalent in the mobile app space. Notable examples include Snapchat [1], Whisper [2], Ask.fm [3], Spring.me [4], RedPhone & TextSecure [5], and Silent Phone & Text [6]. All of these applications offer users additional privacy or anonymity that they would not normally get from a traditional messaging or social network service. For example, Snapchat allows users to send photographs to their contacts that are erased after a set amount of time (1-10 seconds). To view the image, the user must press and hold the screen, and after the timer runs out, it can never be viewed again [1]. As of November 2013, Snapchat users collectively receive 400 million Snapchat images per day [7]. Many critics of the app see it as nothing more than a tool for ‘sexting’, although it’s unclear to what extent it’s used for that purpose [8].

Another popular app (but not as popular as Snapchat) in this category is Whisper. Whisper allows users to anonymously share images with a super-imposed message. These images are shared with all users of Whisper, but all that is known

about the poster of the image is their general location, usually just their state. Other users can then anonymously reply to the image, publicly or privately to the original poster [2]. Based on my use with the app, most posts seem to either be secret confessions or personals (i.e., requests for platonic or romantic relationships). And since users can send private photos to each other anonymously, it can also be used as a platform for 'sexting.'

The two apps I described are, in my opinion, the most interesting examples of mobile apps that promise privacy, but the other apps originally listed mostly involve sending direct messages securely and privately, or posting public messages anonymously. These types of apps clearly pose a challenge to parents who want to protect their children from cyber-bullying and 'sexting', and especially don't want them sending or receiving nude pictures if they're underage [9]. In these cases, law enforcement must be involved, and Snapchat has publicly admitted to sharing user images with authorities when prompted [10]. While this is a real problem, this paper will focus on whether or not these apps live up to their promise of privacy.

To the Community

I became interested in this topic when I was using the app Whisper. I really liked the idea of being able to share images and messages anonymously with the world, and I think this sets it apart from other social networks. However, I was concerned that the app might not be as anonymous as it seemed. Although you only explicitly give the app a username and your location (which is used for sorting posts), providing anonymity is still a hard problem and I had no direct way of knowing if the developers could actually guard my identity against malicious attackers. I started thinking about other apps that

make promises of user anonymity and privacy, and wondered how secure they were too.

When users choose to download an app like Snapchat or Whisper, it's very difficult for them to know how secure their data really is. Customers use these types of applications to share things that they wouldn't normally share (as described above), so they expect that the programmers of the app have taken all necessary security precautions. However, this is not always the case. Other issues complicate also complicate matters further. Theoretically, users read a privacy policy before signing up that legally binds both parties, but anecdotal evidence suggests that few if any users actually read it. Most mobile applications are closed-source, which means that users and user-advocates have sparse knowledge of what security measures are being taken by the app developers. As a result, much knowledge of mobile app security must come from the work of white-hat hackers doing black-box testing of an app. These kinds of independent studies are relatively rare and are usually only done for popular apps, so much less is known about less popular apps like Whisper. Although many apps must go through an approval process before being released (at least on iOS), they don't necessarily ensure that all user data is handled properly. This is a challenging problem for users looking for privacy enhancing mobile apps, but there is no clear solution in sight. And although it's easy to blame the developers, many mobile app developers are working in very small, fast-paced teams and may not have the time or resources to ensure that they're building a secure application. Snapchat was originally launched by a team of two and still has fewer than 30 full-time employees [11, 12].

Action Item

Now I will go over some of the known vulnerabilities in the app Snapchat, and walk through how one of them can be exploited. One of the most alarming revelations about Snapchat has been that all opened images are permanently saved onto the user's phone. This largely defeats the purpose of "disappearing" photos, but the forensic analyst who discovered the archived photos claims that they are extremely difficult to find, which is almost reassuring [13]. Much more commonly known by users is the ability to save screenshots of open images. I know that this is possible on the iPhone, but users of the Android version tell me that app closes when they try to take a screenshot. After taking a screenshot of a Snapchat image, a notification is sent to the sender of the image, which is supposed to act as a deterrent to the behavior.

However, there is another way for users to permanently save an image from Snapchat, except without the sender knowing. This method requires some technical skill, although many apps exist on both the App Store and Google Play store that can automate the process [14, 15]. By giving the app your Snapchat login credentials, it impersonates you and downloads the images while circumventing the Snapchat code that deletes them [16]. Below I will go through the steps I took to permanently save an image I received on Snapchat, without the sender even knowing that I had opened it. I did this on the iPhone, but the process also works on Android. My approach is based on a guide written by Amelia Cuss, and although my process differed slightly from hers, I benefitted greatly from her example [17].

1. *Using mitmproxy [citation] to acquire the encrypted image.*

First install mitmproxy on your computer (instructions here), and send the mitmproxy certificate to your mobile device to enable SSL packet interception (instructions for iPhone here) [18]. This step is important because Snapchat sends its images over SSL. Next, set your mobile device settings to use your computer as a proxy server. On the iPhone, go to Settings -> WiFi and press the (i) next to your current wifi connection. Scroll down to the bottom and turn on manual proxy, set the IP address to your computer's IP address, and set the port to 8080. Now, open mitmproxy on your computer and you will see all packets sent to and from your mobile device.

If you open up the Snapchat app and you have any unopened images, you should see a packet in mitmproxy that looks like this:

```
>> POST https://feelinsonice-hrd.appspot.com/bq/blob
       ← 200 application/octet-stream 78.38kB 1.11MB/s
```

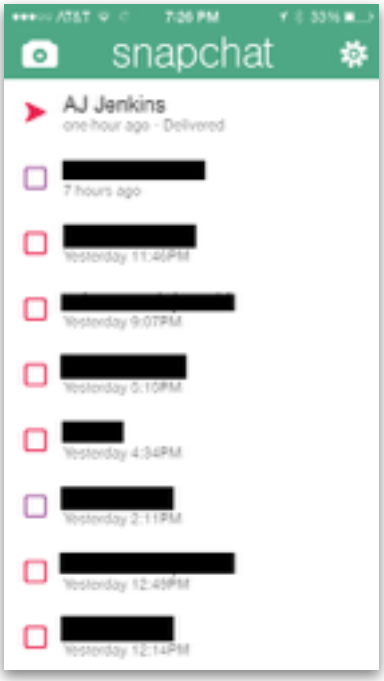
You'll probably get many packets from Snapchat, but you can tell this is the one you want because it's a blob, and it's bigger than the other packets. If you view the packet request, you should see something like this:

```
Request
Response
Host:              feelinsonice-hrd.appspot.com
Proxy-Connection:  keep-alive
Accept:            */*
Accept-Encoding:   gzip
Content-Length:    141
Accept-Language:   en;q=1, fr;q=0.9, de;q=0.8, ja;q=0.7,
nl;q=0.6, it;q=0.5
Content-Type:      application/x-www-form-urlencoded;
charset=utf-8
Connection:        keep-alive
User-Agent:        Snapchat/6.0.2 (iPhone5,1; iOS 7.0.4; gzip)
URLEncoded form
id:         XXXXXXXXXXXXXXX
req_token: XXXXXXXXXXXXXXX
timestamp:  XXXXXXXXXXXXXXX
username:   XXXXXXXXXXXXXXX
```

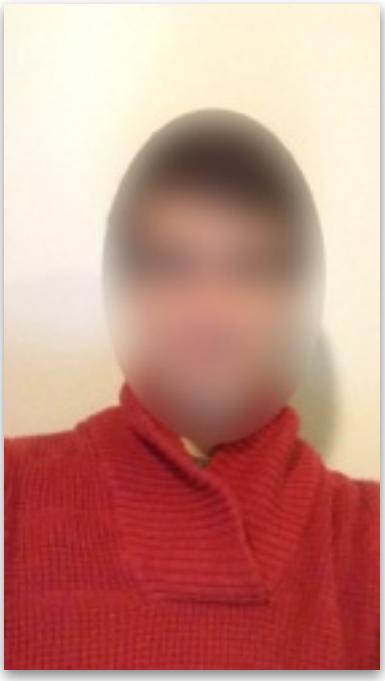
The four values at the bottom have been replaced with X's. This is actually just a straightforward POST request, and you can translate this into a URL and put it directly into a web browser to get the encrypted image. The URL will look something like https://feelinsonice-hrd.appspot.com/bq/blob?id=XXX&req_token=XXX&timestamp=XXX&username=XXX . I didn't need to do anything beyond that, but you can also get the response from mitmproxy. Hit enter on the packet, press tab to switch from the request to the response, and 'v' to open the response in your default text editor. The type of the file you've downloaded is "data", which we will now decrypt.

2. Using a simple script to decrypt the image.

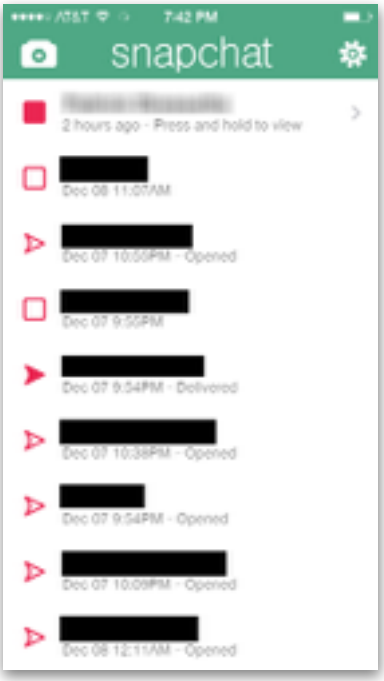
In my supplement to the paper is a Ruby script that will automatically decrypt the image, but here I'll just describe the steps. All images on Snapchat are symmetrically encrypted using AES-128 in ECB mode using the key 'M02cnQ51Ji97vwT4' [19]. Later on I'll discuss my guess for why they used one key for everything. This key is hard-coded into the app, and the source code for the Android version is quite easy to find [citation]. With this information, decrypting the image is straightforward, but you can refer to my code for a concrete implementation. Snapchat images are JPEGs, so once you've saved the decrypted the image and given it the .jpg extension, you should be able to view it with any photo viewer. And that's it! You've successfully saved a permanent copy of a Snapchat image without notifying the sender. Below are my images of the "proof."



Screenshot from the sender of the image. The arrow next to my name indicates that the receiver has not opened the image yet, even though I've downloaded it.



The decrypted image. The sender's face has been blurred.



My screen after receiving the image. The filled in square means that the image has not been opened yet. The sender's name has been blurred.

Discussion

If you followed my walkthrough of the exploit in Snapchat, you may be wondering how an app as popular as Snapchat could have such an easily exploitable vulnerability. My best guess is that the developers have prioritized simplicity, speed, and reliability over security. It seems absurd that Snapchat uses a single encryption key for all users and images, but what would be gained by having multiple keys? The sender and receiver would need the same key, and Snapchat doesn't want to burden users with a complex key sharing mechanism. It seems like a good solution would be asymmetric encryption. This would actually go a long way in preventing images from being intercepted by a third-party, but wouldn't solve the issue of users' ability to circumvent the Snapchat app and permanently save images.

The only real solution is to use code obfuscation in the app to make the key harder to find, and every time you release an update to the app, use a new key and move the key somewhere else in the code. As long as Snapchat updates the key at a faster rate than attackers can find it, this would effectively solve the problem. This general issue of verifying that data from users is coming from your app and not another app or script, is actually a well-known issue in security, without a perfect solution [20].

In general, most mobile apps would ensure better privacy by: sending data over SSL, encrypting all sensitive data, using asymmetric encryption when appropriate, hiding hard-coded secrets with code obfuscation, storing as little information about the user as possible, and releasing frequent updates to patch security bugs. It's hard to say which apps are doing a good job with these things, because issues usually only become known after an attack is publicized or the app is reviewed by an outside security expert.

This problem of determining an app's security is clearly a problem for consumers. There is a demand for privacy enhancing apps, but an app that is easily exploitable may do more harm than good, especially since users are likely sharing sensitive data with the app. One solution for consumers is to simply stop relying on mobile apps for providing privacy or anonymity, but this is not much of a solution at all. Another solution is to only use open source apps that have been verified as secure by the community, but, from my experience, it's rare to see open source mobile apps. Essentially, it comes down to the question of "Can I trust the developers with my data?", and the only way to answer that is by staying educated as a consumer. However, even if you trust the developers, you often must trust the other users that you're sharing your data with. Once you've given another user access to your data, it's almost impossible to control

what they do with it, as the Snapchat exploit has demonstrated. Although the current state of privacy in mobile apps looks grim, maybe one day we'll have technology that lets us trust one another.

References

[1] Snapchat Homepage. http://www.snapchat.com/

[2] Whisper Homepage. http://whisper.sh/

[3] Ask.fm Homepage. http://ask.fm/

[4] Spring.me Homepage. http://new.spring.me/#!/

[5] WhisperSystems Homepage - Creators of RedPhone and TextSecure. https://whispersystems.org/

[6] Silent Circle Mobile App Page - Creators of Silent Phone & Text. https://silentcircle.com/web/silent-mobile/

[7] Blodget, Henry. "The Most Active Snapchat Users Get Hundreds Of 'Snaps' A Day." *Business Insider*. 11/20/2013. http://www.businessinsider.com/how-many-snaps-snapchat-users-get-2013-11

[8] "Is Snapchat only used for sexting? We asked 5,000 people to find out." *Survata Blog*. 2/7/2013. http://survata.com/blog/is-snapchat-only-used-for-sexting-we-asked-5000-people-to-find-out/

[9] Taran, Randy. "Cyberbullying Apps -- Why Are We Allowing Anonymous Cruelty?" *The Huffington Post Blog*. 9/18/2013. http://www.huffingtonpost.com/randy-taran/cyberbullying-apps_b_3941599.html

[10] Munson, Lee. “Snapchat admits sharing images with US law enforcement.” *Sophos - Naked Security*. 10/16/2013. http://nakedsecurity.sophos.com/2013/10/16/snapchat-admits-sharing-images-with-us-law-enforcement/

[11] Colao, J.J. “Snapchat: The Biggest No-Revenue Mobile App Since Instagram.” *Forbes*. 11/27/2012. http://www.forbes.com/sites/jjcolao/2012/11/27/snapchat-the-biggest-no-revenue-mobile-app-since-instagram/

[12] Shontell, Alyson. “Meet The 20 Employees Behind $4 Billion Snapchat.” *Business Insider*. 11/16/2013. http://www.businessinsider.com/snapchat-early-and-first-employees-2013-11

[13] Ducklin, Paul. “Snapchat images that have "disappeared forever" stay right on your phone…” *Sophos - Naked Security*. 5/10/2013. http://nakedsecurity.sophos.com/2013/05/10/snapchat-images-that-have-disappeared-forever-stay-right-on-your-phone/

[14] Snaph-Hack Pro Download Page. https://itunes.apple.com/gb/app/snaphack-pro-for-snapchat/id716560946

[15] SnapCapture Download Page. https://play.google.com/store/apps/details?id=de.innovationz.snapcapture.noroot&hl=en

[16] “How Does SnapHack Work?” *Decipher Media Blog*. 10/15/2013. http://decipher-media.blogspot.com/2013/10/how-does-snaphack-work.html

[17] Cuss, Amelia. “Snapchat: not for state secrets.” *Kivikakk.ee*. 5/10/2013. https://kivikakk.ee/2013/05/10/snapchat.html

[18] Mitmproxy Homepage. http://mitmproxy.org/index.html

[19] Caudill, Adam. “Revisiting Snapchat API & Security.” *Adam Caudill’s Blog*. 12/31/2012. http://adamcaudill.com/2012/12/31/revisiting-snapchat-api-and-security/

[20] “How can I securely authenticate the client application sending me data?” *Information Security Stack Exchange*. 11/26/2010. http://security.stackexchange.com/questions/826/how-can-i-securely-authenticate-the-client-application-sending-me-data