INFO350 – Special Topic

Mobile phones and their security weaknesses

Cameron McLennan
8168187

Table of contents

Every year, mobile phone developers are introducing more and more functions and features into their devices to try and gain more users. While expanding the feature set of a device is good as the device is now able to do more, this also means that there are more possible ways for a mobile device to be exploited. The more that a phone can do, the more it has to be protected. Retired NSA Technical Director Richard George once said "Every piece of functionality is an opportunity for the adversary." (Fink, 2014). This paper would aim to highlight the vulnerabilities in the current crop of mobile phones, going in depth on mobile phone features to see what they do and what the data from them can be used for, exploring how these features make mobile phones vulnerable, and exploring options in how they could be protected. In order to know how smartphones can be security vulnerabilities, this paper will start at the hardware level and work up to the software level, explaining security weaknesses and in what ways they could be exploited.

An example that somewhat outlines how insecure mobile smartphones can be, one of the most powerful people on the planet does have a smart phone, but it may only be "a dumbed down 'toddler' phone". The last President of the United States, or POTUS, Barack Obama was the first president to use a smartphone in the white house, which was a BlackBerry. BlackBerry devices are known to be rather secure with regards to data handling and communications (Manea, 2014), but not even this rather secure OS was secure enough for the NSA's security guidelines. The president was allowed to use BlackBerry but the NSA actually had it modified so that he could only contact his closest advisors (Fink, 2014), and it is unknown whether or not he could call or text from it. In the last year of his presidency, he was allowed to upgrade to an iPhone, but it had also been modified to the point where it could not take photos, make calls or texts, or even play music. Again, the device could be used to contact a select group of people, but it could be used to browse the web, and read news (Moscaritolo, 2017). The current POTUS, Donald Trump, previously used what is thought to be a Samsung Galaxy S3, which was Samsung's flagship back in 2012. This device stopped receiving updates in 2015, so could

very well pose a security threat for the whole of the United States as for those who know how to turn a phone into a bug, it wouldn't be hard to attack an older, less secure device. Trump had allegedly stopped using this device, and has since been given a new, NSA approved device to use instead. It is stated that he has been using his old Android device for tweeting, which is a very large security risk (Moscaritolo, 2017). If he were to click on a malicious link, his phone could become one of the most valuable bugs in the world. If the NSA deem smartphones insecure and have to modify them so extensively that they become what mobile phones back in the late 1990s were capable of, with the only difference being the size of the device, what does that mean for the rest of the population? Are they safe and secure?

Mobile security is very important when there are a lot of users of mobile devices in the world. As of the 8th of October 2017, there is approximately 8.37 billion mobile connections, and there are currently 5.08 billion unique mobile subscribers (GSMA_Intelligence, 2017). This is a lot of people and devices to protect from attackers. The average smartphone in 2017 has a vast array of sensors, sensors that when paired with strong code make some features of the mobile phone work seamlessly, but just because a devices' features work well, it does not mean that the device is secure. As a baseline device for a real life reference point, Samsung's latest flagship device, the Galaxy S8, will be used as a resource for gathering sensors from as it is a popular flagship device in 2017 and will come with many sensors on board, as seen on Samsung's product page (Samsung, 2017). Android will be a larger focus in this compared to iOS, for a couple of reasons, one being that market share for Android vs iOS is almost double, as seen on netmarketshare.com in July 2017 (NetMarketShare, 2017). Another reason will be more exploits available in Android due to various Android operating system versions being installed on many different devices with various hardware combinations compared to iOS. People who have the drive to attack mobile devices will want to target as many people as possible, so if they target the Android OS they will be targeting more users. Android's wider spread will mean that there are simply more ways that a device can be vulnerable, having to support all the hardware differences and any skins on top of Android can also provide weaknesses. (NetMarketShare, 2017)

Sensors

First off, a commonly used sensor would be the Accelerometers (Mishra, 2017). The basic idea of an accelerometer is that it measures acceleration that the handset undergoes. In mainstream devices, such as the mobile phone, they are used for orientating axis, which can in turn be translated to whether or not the contents on screen should be in portrait mode, or landscape mode. It is designed to determine the orientation in the X, Y and Z axes. Exploiting this sensor would result in being able to tell if the device is in place, such as sitting on a desk or in a pocket with the user sitting down, or even in the hand. It could help tell the exploiter if the user is moving if they can track the live accelerometer updates and track a pattern which would occur if a user had their device in their pocket when walking, or even in their hand when walking. (SmartfoneArena, 2016) (Google, n.d.)

The gyroscope performs a similar function to the accelerometer, but the gyroscope measures angular velocity along one rotational axis (Mishra, 2017). However, they can also be set up to be able to register tri-axial movements. The gyroscope is used in applications such as googles sky maps, and can be used in tandem with the accelerometer, depending on what the

task desires. If one were to exploit this sensor and utilize data from it, they would find that not much useful data can come from this sensor that could be used maliciously. (SmartfoneArena, 2016)

The magnetometer sensor, or geomagnetic sensor, within a smartphone is designed to detect the Earth's magnetic field, and utilizes an effect known as the Hall-effect in order to detect the magnetic field (Mishra, 2017). It can also be set up in such a way that it can measure the X, Y, and Z axes. The sensor produces a voltage which can then be converted into a digital signal, which represents the magnetic field strength and polarity (RotoView, n.d.). Another tech used for magnetometers measures change in resistance, which is based on the changes of the magnetic field. This sensor could be used in order to track a path if it is set up to measure the changes in the X,Y and Z axes, but a better way to track someone would be to use the GPS and a barometer. (SmartfoneArena, 2016) (Google, n.d.)

The proximity sensor is an infrared light beam that is used to detect if anything is in front of the screen and reacts when conditions are met for it to react, such as when a user of a smart phone accepts a phone call and holds the phone to their ear (Mishra, 2017). The screen will turn off, due to this sensor, in order to eliminate any accidental touches that could occur from the ear of the operator. This sensor could be exploited in such a way to say either if the user is on their phone making a phone call, but even then they could be using their device on speaker mode, meaning it can't give a yes or no as to if the user is even talking on the phone. This is because sensor requires you to be close to get it to register. This wouldn't be the best sensor to try exploit.

The ambient light sensor is designed to measure ambient light and can adjust screen brightness accordingly, in order to reduce eye strain and unnecessary screen wear due to constant high backlight brightness. This sensor could actually have some use if it were to be exploited as it could tell the exploiter if someone was sitting in the dark on their device, or if they were in a well-lit location. (Mishra, 2017)

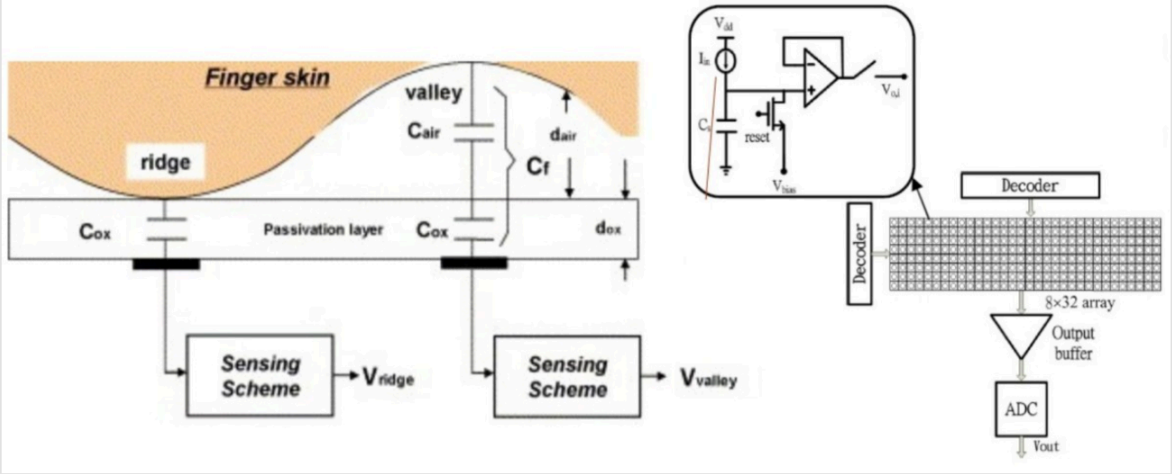
The barometer is a sensor which is designed to measure atmospheric pressures (Mishra, 2017). It can be useful in situations where you are unable to get a data connection. For instance, the Galaxy S8, Samsung's current flagship device as of July 2017, has a barometer in it. Apples devices also have barometers in them. The barometer is designed to measure atmospheric pressure of the area around the device, so this sensor would be able to tell someone if the device and user were climbing in altitude or way down low. (Levi, 2011)

Smartphones, or phones in general, should all contain at least one thermometer or temperature sensor (Mishra, 2017). The reason for containing at least one thermometer would be for the battery temperature. Batteries can combust or even explode, thus warranting the need for a temperature shut off, with the trigger being a temperature threshold measured by a thermometer sensor within the device. There have been devices, such as Samsung's 2013 flagship smartphone, the Galaxy S4, which did have an ambient temperature sensor. This sensor enabled the device to measure the temperature of the area around it (GSMArena, 2017). One would imagine the accuracy of this could be skewed by variables, such as device temperature or materials of any case that is on the device. This type of sensor would be good should you not have any data connection but wish to get at least an estimate of the temperature of the surrounding area. This sensor wouldn't be a very useful one to exploit for a couple of reasons. One, there isn't always an ambient temperature sensor on the device, so the only temperature sensor(s) on board would be to monitor hardware, such as the battery, or the SoC

temperature. Another reason that the sensor wouldn't be the best to try and exploit is the fact that if the device has an ambient temperature sensor, its readings could be skewed by the heat of the device, the user or where the device is stored. This would mean that any reading attained from the sensor, if possible, would be most likely an inaccurate result for the desired outcome, rendering it useless to exploit. If the hardware monitoring temperature sensor(s) could be manipulated on the other hand, this could endanger the device and potentially the user as the sensors are installed to provide overheating protection for both the hardware and user.

Some smartphones these days include a heartrate sensor (Mishra, 2017). It uses an LED to pulse light against the finger, and then there is an LED sensor to detect the light that gets reflected back. These light reflections vary as blood flows through your finger. Usually these are placed on the back of the device near the camera, where one will naturally rest their finger to support the top of the device when holding it with one hand. This is a sensor that won't get used a lot, so if it were to be exploited, the data readings could potentially be sporadic. There isn't really a use to exploiting the heart rate sensor on a mobile device. (Samsung, n.d.)

A recent sudden popularity has risen for the fingerprint sensor (Mishra, 2017). This is a sensor that users can use to quickly unlock and start using their device. There are two major styles of fingerprint sensors. The capacitive sensor and the optical sensor. (Android Authority, 2016) The capacitive sensor is essentially a bunch of capacitors which detects the groves of the fingerprint when a finger is placed on the sensor as the ridges and gaps in the print are what gives the capacitors a charge value. The data that the capacitors get is then compared to patterns stored on a dedicated integrated circuit, which will then process the scanners' data to see if the current user is authorized to unlock the device, and then either rejecting or accepting the access request. Over the past few years they have come a long way in mobile phones.



The theory and architecture behind a capacitive fingerprint scanning chip.

(Android Authority, 2016)

The other type of sensor that is commonly found is the optical sensor. It is designed to take a 2D image of the fingerprint, and using the light and dark areas of the image to gather what the shape of the print is. When Samsung first brought their fingerprint sensor out in the Galaxy S5, the user had to swipe down the sensor. This is because it used an optical image scanner to scan and rebuild the fingerprint image inside of the device, which was a bit fiddly

for users to use and didn't work as good as current sensors, which are all press and hold style, with no swipe needed. Nowadays, most, if not all sensors are a hold to unlock, with some sensors working when the device is asleep, working as a wake button. This is a sensor that is tasked with protecting a user's data, so if there was any sensor to try and exploit, this could be a good one to try exploit. It has previously been exploited, as seen on the YouTube channel Tested, the iPhone's fingerprint sensor was defeated with a molding of a finger, and some graphite to give the finger capacitance, and then scanning the finger. (Tested, 2014)

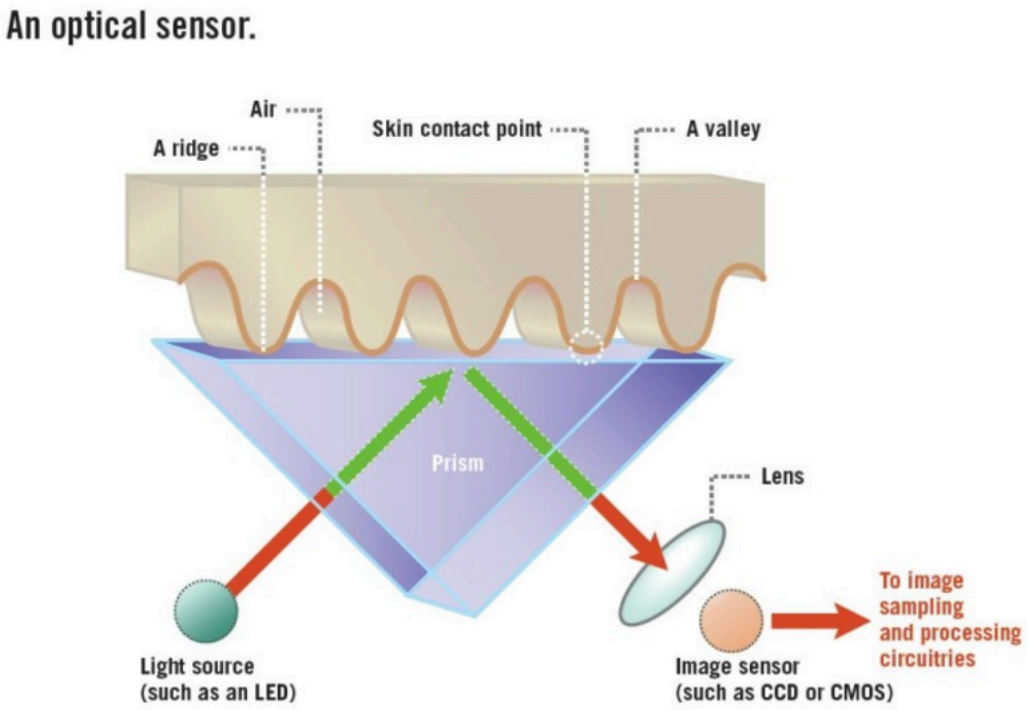


Figure 2

(Android Authority, 2016)

All smartphones will have to have a microphone, as without a microphone, they can't be a phone. A phone is first and foremost for audio communication between two or more parties, and that requires a microphone (Mishra, 2017). Most smartphones have a second microphone implemented for the sole purpose of noise cancelation, which attempts to cancel out background noise in order to try send a clearer voice transmission when in a loud area. This would be a good part of the smartphone to exploit. Being able to hear what people are directly saying is invaluable in some situations. For instance, if you were involved in negotiations over something worth a lot of money, knowing what the other party are thinking is information that could go a long way. You could leave a phone in the room that you are doing the negotiations in, and say you wish to step out with a colleague for a moment. You could then ring the phone left in the room and have it answer automatically, something you would have set up prior. A blatantly placed bug, but in modern times where everybody has a smartphone, one that would go completely over looked. The only way to completely stop something like this is to jam all signals going into and out of the room. Not impossible, but an investment in itself.

Most smartphones in 2017 have two image sensors, or cameras, one on the back for taking high quality images of whatever is behind the phone and one on the front of the device.

Smartphone cameras have improved quite a lot over the past few years, with the quality of the the rear facing camera's image rivaling, or even surpassing that of point and shoot cameras. The second sensor, the one located on the front of the phone, is usually positioned up the top of the device beside the ambient light sensor, proximity sensor and the speaker used for phone calls. The point of this camera is to capture images of the user, or in some cases, capture images of those trying to get into the phone by taking photos when an incorrect password is tried multiple times. The front facing camera can also be used for facial recognition or iris recognition, should those methods of protection be implemented into the device. Taking advantage of this sensor could lead to some interesting results. You could build the world around the user out of images as they use their device. This would be a very useful sensor to exploit. For example, should a user be at home, you could see what valuables they have in their surroundings, you could see points of entry, or even get the layout of their house. If geotagging or geocaching is enabled on their images, you could also get GPS coordinates of their location embedded in the image, so you know exactly where they are. If geotagging is not on, if you had the ability to exploit this sensor there isn't any reason why you couldn't turn geotagging on. On the flip side, this could also be used for good. It could help anyone who is looking for you to be able to see what is in front or behind the device, and even get the geotag information to see where you are. It is unlikely that this sensor will be used for good like that though, as it is an odd way of going about finding someone.

There are a few smartphones in 2017 that are equipped with an Iris scanner. They are designed to protect a user's device from others by not allowing access to the operating system unless it can scan a user's eye and verify that this user is in fact allowed to use the device. The way an iris scanner works on a mobile phone is a beam of infrared light is projected towards a user's eye, and the front facing camera on the phone has a special filter that can capture the reflected infrared light, which will contain details about the pattern of the iris of any particular user. (Arici, 2017)

Smartphone Connection Methods

Now that some major sensors have been covered, some more key components of the smartphone will have to be discussed, as these are some of the most compromising features of smartphones, as they are the features which allow data transfer on and off of the smartphone.

A major feature that all smartphones will have will be Wi-Fi connectivity, as a smartphone usually requires the internet to get access to applications that take advantage of sensors to give the user data, or to update software in order to protect the device from any current software flaws. Currently the standard for Wi-Fi is the IEEE Standard 802.11-2016 (IEEE Standards Association, 2016). The Wi-Fi connection would be one of the most used features of a smartphone in 2017 as so many people are dependent on the internet to use services such as Facebook, Instagram, YouTube, or even just using their Wi-Fi access to web surf. The Wi-Fi connection on a smartphone is one of the biggest weak points of the device, as the data throughput is rather high for modern smartphones, so a lot of data could be loaded in a relatively short timeframe, thus allowing large, malicious files to be loaded and potentially vast amounts of damage is able to be done.

Another connection that a smartphone will most likely have is mobile data connectivity as if it has the cellular component which makes it a phone, the mobile data is data transmitted through the cellular network. This connection type is good for when a user is away from their Wi-Fi connection but need a form of internet connection as all they need to access the data portion of the cellular network is an allowance of data and cellular coverage. Many places now offer what is known as 4G, which is the current latest standard of data transmission over a cellular network. It is rated at peak speeds of 100Mbps, but many places both locally and internationally cannot get anywhere these rated speeds, with their speeds ranging from 30 – 60% of what they should be to meet the standard. While they don't completely meet the standard, the speed is plenty fast for doing anything you would want to do while not at home. The cellular connectivity can help emergency services locate a user's device, and hopefully the user as well. They can do that by triangulating your location between three cell towers. From the ping, the time taken for each ping to get to the device and back to the tower is known, and from there a rather accurate location can be acquired (iiiweb.net, n.d.). There is also another option, which works in a similar way. This option is that of what is known as a "silent" text message. The silent SMS's real name is a Short Message Type 0. The GSM technical specification states that "the handset must acknowledge receipt of the short message but may discard its contents" (T, Did you know that "silent" text messages can be used to track your whereabouts?, 2014) (GSM Technical Specifications, 1996). This means that the sender gets a notification that their message has been received, but the receiver doesn't know that a message has come in, or been replied to. This can be used to test if a phone is on or not, but as stated before, can be used to track locations of a device, along with the help of information from cell towers. These type of applications are available from the Google Play Store, and Apples App Store, but wouldn't be as good as what law enforcement agencies could have access to (T, Did you know that "silent" text messages can be used to track your whereabouts?, 2014).

The next connection is the Bluetooth connection, which is a popular form of connection on the modern smartphone. The current IEEE Bluetooth standard is 802.15.1-2005 (IEEE Standards Association, 2010). It is used in order to interface with many items in the world, such as car stereos, portable speakers, smartwatches, and many more items. It was born in the mid 1990s, and uses UHF radio waves in between the 2.4-5ghz spectrum. Class 3 Bluetooth radios have a range of around 1m, class 2 radios have a range of around 10m, and class 1 has a range of around 100m. Smartphones use the class 2 Bluetooth specification. The current version of Bluetooth in flagship devices is version 5, with its maximum data transfer rate being around 50 Mbps, double the speed of the previous standard of Bluetooth. It is technically double the speed because the data transmission speed was doubled. This is also applied in low energy mode, with its maximum data transfer rate being 2 Mbps. It also has a maximum range of 240m, quadruple that of the previous standard. The Bluetooth connection is a connection that can be exploited as malicious items can be loaded onto a device with a Bluetooth signal on, or data could be pulled off the phone via the Bluetooth connection. While it is a convenient connection, it does have its share of risks.

The last connection is the NFC connection, which is designed to have the smartphone get close to some form of a scanner which is looking for or waiting for a signal to appear and then react accordingly. It is similar to how RFID works, but where RFID only reads, NFC can both read and write. NFC is an emerging piece of tech, whereas RFID has been around for a while. NFC is convenient for sharing data by having it as an authentication step in a data transfer process, where users could touch phone backs to authorize the connection and data transfer, and use Bluetooth to connect and transfer data, as Bluetooth has a larger range. NFC is growing in

popularity with companies like Apple and Samsung pushing the idea of "contactless payment", which is where a user registers their bank card(s) to a device and they can then use their smartphone as a "virtual wallet", holding all their cards in one place. This usually requires a security step, like a fingerprint, or a passcode to authorize the transfer. This connection method could also be abused rather significantly, similarly to how one could have their money stolen off of their contactless payment credit card. Someone who wished to exploit this connection could use it to authorize a Bluetooth connection, and then start to download a copy of a user's device over the connection that gives them higher data throughput over a longer range.

Device Lock Screen Protection

The first line of defense in a smartphone is usually the lock screen. There are many different methods to protect the smartphone from anyone who isn't that serious about getting into it. There is always no lock on a phone, which is always the worst for security. It is like leaving the bank vault door open and waiting for someone to discover that it is open and take advantage of it. Any lock is better than no lock. From here, there are many forms of locks, but strengths of these locks are not all equal. The different types of locks would be: passcode or pin code, password, pattern, facial recognition, iris recognition and fingerprint recognition.

The pin code or passcode, is a common protection method on smartphones. Usually the pin code is also used as a backup for other security methods, such as the fingerprint recognition lock or iris recognition lock. The length of the code is usually at least 4 digits, but there is some software that will let a user have many more than 4 digits for the added security. For a standard 4-digit pin, there can be up to 10,000 different pin code combinations. The more digits that can be used the stronger the protection is. For Android, the actual pin string is run through an encryption algorithm which encrypts the pin and what is stored in the /data/system/ location is a salted SHA-1 hashsum and an MD5 hashsum of the pin (Nelenkov, 2015). Because the pin hash is also salted, it isn't very viable to use a dictionary attack. SHA-1, or Secure Hash Algorithm 1 is a cryptographic hashing algorithm which produces a 160-bit hash value (Rapid Web Services, 2017). A dictionary attack is an attack where a dictionary of possible values for a specific task is created and used to either get into (such as a brute force attack) or for encryption, such as LZW encryption (Rouse, n.d.), (Duke University, n.d.). For iOS pin code storage, they use a technology called Data Protection. It creates a new encryption key, separate from the devices master cryptographic key, which is a mix of the devices unique code, and its line code (e.g iPhone 5, 6 etc) and entropy, which ensures that each device reset results in a new master cryptographic key. The device unique code and line code are hardware based. All files that get encrypted are encrypted with a key that gets derived from the master, and because this is all hardware based, it is very quick. When the new encryption key is created for the passcode, it is used to encode certain files that are marked as "critically important" by the OS. The passcode itself actually gets used as part of the encryption key, but after the device locks it gets discarded. iOS cannot decrypt any data when the device gets locked as the encryption key has been discarded. Because it is only temporary, and doesn't actually exist on the device, Apple cannot help users recover their passcodes (Tabini, 2013). There is no perfect solution to being able to have your passcode recovered easily and still maintaining security, unless the OS has security features that can allow you to link a device to an account that will let you recover the devices passcode or password, should you forget it. Even this isn't as secure as only having the one way into the device, as as little ways in means less weak points in the devices security.

The password can be a very strong method of protection as usually it is a string of characters that can be quite long, but most users won't like to have to type a word out each time that they wish to access their device, as something like the pin code has a much easier interface to use as it only has a 3x4 grid of buttons to worry about hitting, whereas the keyboard is the full on screen QWERTY keyboard. This sizing difference will mean that the amount of errors that occur when typing in a pin as compared to a string of characters would be lower for the pin as there is less to hit, and more space per on screen button. This could also not be as easily attacked as the pattern lock with its smudge attack, which will be talked about later. The strength of the password gets exponentially better going off of Password Depot's results. % characters, 3 lower case and two numbers takes 0.03 seconds to crack, but going up to 9 characters with 2 uppercase, 3 lowercase, 2 numbers and 2 special characters results in a time of 9.1 years to crack (Password Depot, n.d.). Special characters appear to make cracking a lot harder. With regards to storage of the password, Android stores the password in a special file by the name of password.key, and is stored in /data/system/. As with the pin, the actual password string is run through an encryption algorithm which encrypts the password and what is stored in the /data/system/ location is a salted SHA-1 hashsum and an MD5 hashsum of the password (Nelenkov, 2015). In order to attack the password protection method, you only need to attack the weaker hash, the MD5 hash (Nelenkov, 2015). iOS does not appear to allow users to set a password as protection for their device, only a passcode and fingerprint protection on currently released devices.

The pattern lock screen protection another method is not found on iOS, but is found on Android. This method of lock screen device protection is where you connect at least 4 points on a 3 x 3 grid of points to choose from, but each point can only be selected once. This, along with the password or passcode protection has increased strength if more digits, characters, and in this case points, are included. The possible combinations that can come from the pattern lock is 389,112 distinct pattern combinations. While this is still a few combinations, they can be more easily defeated compared to passcodes with an attack like the smudge attack (University of Pennsylvania, n.d.). The Smudge Attack is basically utilizing the oily residue left on smartphone screens by a users finger to see the direction of continuous swipe needed to unlock a device protected with the pattern lock screen. As for storage, Android stores the pattern in /data/system/, under the name gesture.key (Spreitzenbarth Mobile Security and Forensics, 2012) (Nelenkov, 2015). This is also the location of the password storage. Given how foolish it is to store any key that is meant to be protecting data in plain text, the pattern is stored as an unsalted SHA1-hashnum of the pattern (Spreitzenbarth Mobile Security and Forensics, 2012). The 160 bit hash value is known as a message digest (Rapid Web Services, 2017). This hash value is usually rendered as a hex number, ending up at 40 bits long. SHA-1 is not considered strong against those who have the resources to throw at the algorithm to crack it, but for the vast majority of people it is sufficient. There is a worse problem for Android, and that is the issue that given a user cannot use the same point twice, meaning that there is a finite amount of combinations that a user could use. This means that a dictionary (list of all possible values) could be made easily, and given the data is unsalted (not adding a random key that is stored with the hash so that no two patterns result in the same hash), it means security is weakened even more (Spreitzenbarth Mobile Security and Forensics, 2012).

Facial recognition as a security feature is an interesting one, as originally it just mapped the users face with the front camera, and if the image matched it would let the user in. This used to be easily circumvented, as all someone had to was was hold a picture of the user who was registered to unlock the device in front of the camera (GdD, 2013). Updates to this software made a requirement for it to do a "liveness check" to check that it was not just a static image,

and checked for some form of blinking to reduce the ability of thwarting this security method. With the recent announcement of the iPhone X, Apple decided to get rid of the home button which means for the iPhone X, it won't have fingerprint protection. It does however have facial recognition now, termed by Apple as Face ID. The way that Face ID works is it uses an infrared camera to create a 3D model of the user's face. This is said to in theory to "prove more nuanced than previous two-dimensional systems." (Finley, Can Apple's iPhone X beat Facial Recognition's Bias Problem, 2017). The iPhone X also comes with a "flood illuminator" which is an infrared light which floods the users face with infrared light so that the facial recognition can work in light and dark settings. After this step, a 30,000 point infrared laser matrix is beamed out, which reflects off of the flood light. A special infrared camera is then used to detect subtle changes in the previously mentioned laser matrix reflection, and any small movements that your face does without you noticing. This is what enables the cameras to capture accurate 3D depth data of the users face (Triggs, 2017). This tool is actually also used in one of Apples software features called Animoji, which takes a users facial movements and puts them into that of emojis, so users can send friends emojis that talk and move their mouths as a human would, with the senders voice talking along with the emoji. As Samsung does with their facial and iris scanning and processing, the data is processed on an isolated and protected chip on the device, and is also stored there as it is one of the safest places for that type of data, and it minimizes data handling so that nothing could potentially intercept it if it were to be transferred to main memory. Google also do this, and more will be discussed about the TEE, or Trusted Execution Environment later on. Because depth can be attained via the use of infrared light and cameras, it will be much harder to thwart as compared to previous facial recognition methods, as they merely rely on the front facing camera, and some form of liveness check. This also means that it will be much harder to get past, as while the iris scanner of Samsung's devices have been thwarted (more on this later), getting the different depths of a user's face could prove to be hard. This is quite possibly the most advanced facial recognition technique available in smartphones as it uses more than just a standard image to recognize the user. Previously, facial imaging has had troubles with identification of people with darker skin tones. As explained in Finley's article, it is a problem that has been known to exist and still exists with some forms of facial recognition as Finley states "In 2015, web developer Jacky Alciné tweeted a screenshot that showed Google Photos labeling a picture of him and a friend as gorillas." On the twitter image thread it appears as if it caught the attention of developers who could do something about the problem, and were able to deal with it (Alciné, 2015). Another face detection feature which was mentioned as being demonstrated at the iPhone event was part of its automated portrait-lighting mode, which was working with a variety of skin tones. So maybe Apple has managed to crack the code and figure out how to deal with the problem of skin tone and facial recognition in order to create a usable experience for all users. If they did manage to get it working well, this would mean that other companies could get proper facial recognition working for a wider variety of uses in the world. While this method does sound promising in theory, its strengths and weaknesses will not be able to be seen until consumers can access the device and fully test the capabilities of this device protection method.

The iris recognition lock for smartphones is starting to appear on flagship devices, such as Samsung's Note 7 and the Galaxy S8 and S8+ devices, and like the facial recognition when it first appeared, it has already been defeated. As stated previously, they use Infrared light and the front facing camera in order to detect the eye, and the specific features of the eye and then compare this to a database of iris patterns that are allowed into the device. They have been defeated by using a high resolution image of the eye, a laser printer and a contact lens. The contact lens is so that the image of the eye can be given the contour of the human eye, which in some way could be considered a liveness test as without the natural shape of the eye which

the lens provides, you can't trick the system. There was another form of iris scanning for security on mobile devices, but by a smaller company. ZTE had a project by the name of Hawkeye, which was rumored to be based off an earlier project by the name of Eyeprint ID but unlike Hawkeye, Eyeprint ID did make it to some of ZTE's devices. Eyeprint ID used the phones front camera to identify blood vessel patterns in a user's eye, which much like the iris, retina or fingerprints, are unique to each person (Triggs, 2017). While this was a good idea as it was effective and cheap, it is inferior to using infrared light to scan faces and eyes for security as a just a 2D image protection process can be fooled. This isn't to say that infrared scanning based protection methods are perfect, but they are superior to standard image based protection methods.

The last lock screen security method is the fingerprint lock, which has been defeated, but as with defeating the iris lock security method, it requires a bit of effort and a bit of careful planning to actually defeat. The fingerprint is becoming a more common method in protecting a device these days as it is rather quick and convenient, given that some fingerprint sensors both wake and unlock the device in one motion, and the fingerprint sensors location on the device is another convenience point if the sensor is where a finger would naturally rest on the device. Pre Android Marshmallow (6.0), most vendors who were implementing fingerprint sensors all had different ways of using the scanner and storing the data captured by the sensor. Post Android 6.0, Google have implemented their fingerprint sensor API, and have developed something the Trusted Execution Environment (TEE). The security strength of this protection method is rather strong given how unique a fingerprint is, but is weak if someone really wants to get into your device and you protect it via fingerprints. You can actually delete files in the /data/system/ location on the smartphone and it will clear the passwords and whatever else is stored here out, but if there is a fingerprint enabled it will still be enabled and protecting the device. This is because the fingerprints are stored in a "tamper proof" integrated circuit inside of the smartphone. This chip is isolated so that the OS, apps and API calls cannot access your fingerprint data. The chip that is actually storing the data is an ARM chip. There isn't any software way to access the fingerprints without accessing the hardware chip directly. Should the chip be tampered with, it should disable itself in order to protect the data stored on it. Given that this IC has its own storage, processing CPU, memory, and firmware, it must be sealed off so that nothing can capture any radio frequencies that the chip gives off, as this could be a weakness which could lead to the fingerprint data being read. The chip stores the data securely through the use of its Trusted Execution Environment based TrustZone tech (Qualcomm, n.d.). Each time the fingerprint gets scanned, it creates an authorization token and compares it to the stored token. This is also used so that the fingerprint sensor can directly communicate with the chip storing the print data, so that no software can capture the fingerprint data during any step of this process. Qualcomm builds this into its Secure MSM architecture, with anything not running a Qualcomm SoC using the ARM TEE (Qualcomm, n.d.), or Secure Enclave, as referred to with regards to Apple (Claburn, 2017). It all is the same idea of storing the data on a secure portion of the or a processor that the OS doesn't have regular access to (Liu, 2016). Using fingerprints to log in to websites or purchase items online has become possible without compromising biometric security through The Fast Identity Alliance (FIDO), who have developed secure and strong cryptographic protocols which can allow password-less authentication between the hardware storing the biometrics and services that would use it to authenticate purchases or logging in, as mentioned previously (FIDO, n.d.). All comparison and processing in general that has to be done with regards to fingerprint security is done on chip, with the actual image being captured with some form of scanner, usually a capacitive scanner (explained above). The scanner takes in the data for the algorithms to break down and make sense of, and the key features that they look for is where ridges and lines end, or where

a ridge splits into two. These unique features are called minutiae. For a match to occur, several of these minutiae have to match the stored reference prints minutiae. For the stored reference print to be compliant with the operating systems guidelines, the fingerprint must be cryptographically authenticated. It has to be signed by a private, device specific key, and other data (absolute system path, finger ID and group), which is unique to the device itself. This means processing power can be reduced and unlock speed can be increased. This technique also helps if the fingerprint is off angle, maybe even completely upside down. Applications that utilize fingerprint identification never see the print, they only see if authorization has been given to allow the user to use the application (or a part of it) or not.

Android OS

(Current latest Android revision for reference: Android 7 Nougat)

With Android ending up on millions upon millions of mobile devices, with multiple manufacturers producing their own skinned versions of Android, it is no wonder that the Android OS is largely fractured in comparison to iOS. Apple only have a handful of devices to support and know the exact hardware makeup of every device in their eco system. They can choose when to stop supporting a device and you will be well aware of it, but Android is a bit different. There is a vast amount of different hardware configurations running Android in 2017. It is a bit harder on Android to figure out if you will be getting the latest and greatest software as if you aren't running a device from Google's own device line up or aren't on a flagship device from the likes of Samsung or Motorola, you are in the dark a bit with regards to your devices software future. With iOS, it is easier to find out if your device will be getting an update, or is still going to be supported or not. The skin on top of Android that many vendors are running on their devices need to be carefully implemented and bug checked when Google pushes software updates of any size, whether it be a simple security patch or a whole new operating system.

The website www.source.android.com has a security bulletin section, which collates and lists all the bug fixes and states security threat level of each bug, who discovered it and when. In the notes month of January, patch level 2017-01-01, the vulnerabilities were: two counts of remote code execution vulnerabilities, three counts of elevation of privileges, and three count of denial of service (DoS) vulnerabilities. The remote code execution is quite explanatory in itself, as the vulnerabilities provided a way for code to be remotely executed on the device, which could be damaging to the contents on the device or could be used steal data off of the user through the use of a key logger. The denial of service vulnerabilities were vulnerabilities which allowed a specifically crafted file to be loaded and remotely executed by the attacker to cause the device to hang or reboot. This is particularly bad if the attacker can attach the malicious file to an action that is common when using the device in order to constantly cause the users device to hang or reboot on them. One of the denial of service vulnerabilities was discovered in core networking, and would allow an attacker to use a specially crafted network packet to cause the previously mentioned hanging or rebooting of the device. Given that being able to connect to wireless networks is a major function in smartphones, this vulnerability could be a crippling vulnerability for a mobile device. There have been some instances in the past where iOS devices have actually been susceptible to receiving text messages with certain characters that actually caused the device to hang and reboot, but this will be expanded upon later. The other security vulnerability from the month

of January that was fixed was some privilege elevations. Privilege elevations in some cases allowed malicious applications to "execute arbitrary code within the context of a privileged process." This was an issue which gave capabilities to third party applications that they usually shouldn't have. Another issue that was addressed in the January bulletin was that of information disclosure. This vulnerability is where some applications could access information that they should not be allowed access to without permission.

Most of these vulnerabilities were discovered in between the months of September through until November, and were deemed to be fixed and this fix being pushed in the 2017-01-01 patch. This small list of bugs and explanations are just some of the bugs discovered that affect the Android system as a whole, with the rest of the 2017-01-01 bulletin page discussing the more specific security vulnerabilities discovered, which got amended with this patch. These bugs were to do with things like vulnerabilities in Qualcomm components, a previously discussed problem, Nvidia driver issues and MediaTek driver issues to name a few. Most of these vulnerabilities fall under the "elevation of privilege" problem tree, but some, such as the Qualcomm component vulnerabilities were rather serious as they have been talked about since way back in 2015 but really only got fixed as of this patch, with Qualcomm themselves being the ones who discovered them. Windows Mobile OS is not that popular, as seen on Net Market Share (NetMarketShare, 2017), as it has a sub 1% total market share for the month of July, but it is still a top 3 mobile OS. While this is not good for Microsoft's profits, it is good from a security standpoint. Because not many people have Windows Mobile OS devices, attackers will be less likely to attack these devices and people as there will be less chance of them getting something back from such a small user base in comparison to putting time and effort in to attacking an OS like iOS and Android, especially Android, given how many different in use versions of Android OS is out there. Although, just because it appears secure because not many people have found bugs, doesn't mean that it is actually secure. It could still suffer from bugs that are merely undiscovered.

Windows OS Security

Another operating system that is somewhat well known, although not the the most popular OS, is that of the Windows Phone OS line. The Windows Phone OS was released as a successor to the Windows Mobile line. Windows Phone OS began back in late 2010 with windows 7 Mobile, which had some iterations upon it before a next major step in OS release, as Android or iOS have in their OS revisions. One large update, and over a period of months smaller updates for the initial large update come out, until the next major update is ready. After Windows Phone 7, it went to 8. It then went to 8.1, and is currently at Windows 10 Mobile. These updates are following the desktop update cycle, which had exactly the same updates of 7, 8, 8.1 and currently is on 10. Microsoft was, and still is aiming for a unified ecosystem, with universal applications. These universal applications will be written using a common code so that the applications can work on all manner of devices that Windows 8.1 or higher can run on. iOS and Android are not able to do this (Chacos, 2014). Because of the unified ecosystem, developers are able to reuse a majority of code with only minimal code changes for platform changes. This is possible as Microsoft has essentially used similar code for their OS on mobile and desktop, with the smartphones that run Windows Phone OS of some form having a more mobile phone friendly UI to work with. By doing this, and how Apple have their ecosystem with iOS and OSX cross platform integration, Android is the furthest behind the curve in terms of having a cross platform ecosystem as both Windows Mobile and iOS have an OS that they can integrate well with. Android doesn't actually have any OS, but it does work best with

Windows. The devices could be susceptible to bugs such as the "broadpwn" bug that plagued Android and iOS devices alike as it was a bug in a third party piece of code. It depends the developers of the device decided to use the affected chips or not. This bug will be further explained later in the paper. While it may be safe from specifically developed attacks for the mobile OS, it may still be susceptible to attacks which are designed to attack the desktop OS, as the mobile OS shares a good bulk of its codebase with its desktop OS brother. This can be somewhat prevented though, with the use of something called "sandboxing" (Bates, 2016). Sandboxing is a security technique where an application is run in a container of sorts, where the application has limited access, if any access at all, to personal files or folders. This is good for testing an application to see if it contains suspicious code as it is a virtually isolated environment (Khanse, 2017). As mentioned before, Microsoft has its eyes set on the universal applications. In the implementation of the framework for the universal applications, Microsoft has limited the API platform for actually building these applications in such a way that these universal applications have to run inside of a sandbox, which means that they cannot get to other applications. This API implementation is similar to how most smartphone applications are actually built, but it makes it harder for developers to develop applications that are robust productivity applications (Weinberger, 2015).

iOS OS Security

(Current latest iOS revision for reference: 11.0.1)

While Android is a wider spread, harder to secure OS due to how many hardware and software combinations that it is running on, iOS does not escape its share of bugs, flaws and general OS security problems. In recent years it seems as if iOS has suffered from some rather interesting and unique bugs. There have been instances where if a specific string of characters is sent to an iOS running device, it can cause the receiving device to crash and reboot, or lock users out of the Messages application (Grenoble, 2015). One of the strings of text which caused the crash is as follows:

effective. Power لُلُصّبُلُلصّبُررً ॣ ॣh ॣ ॣ 冗

While this was in 2015, there was another instance of users being able to crash other users' devices in 2016, but this time it was using a different method. The idea behind the other method was all the attacker had to do was to send a "large contact file" via iMessage, and when the recipient clicked on the message, the message would be too large and complex for the CPU to deal with and will crash the messaging application (Hein, 2016). These are both forms of denial of service attacks, and should not be performed by anyone.

There was also an instance where sending a some emojis to an Apple device on a specific OS revision would cause the device to slow and eventually crash. These emojis were "a waving white flag emoji, a zero, and a rainbow emoji" (Eadicicco, 2017). The device ends up becoming unusable for a short time, around about a minute according to the Telegraph (Tltcomb, 2017), but YouTube channel EverythingApplePro states that the time period is actually more like three to five minutes (EverythingApplePro, This Text Will CRASHANY iPhone , 2017). These kind of bugs are large security flaws in the operating system on all of

Apple's devices as there is only one operating system that the iPhone runs, with the only difference being the revision as there are no skins that get applied to the iOS OS, it is what it is, which is like Android without the skin(s) applied by other vendors. For example, on an Android device, the Galaxy S8 has the operating system as Android but the skin is TouchWiz, Samsung's skin of choice for Android. There doesn't appear to be these kind of bugs in the Android operating system, and it is far more fragmented than that of iOS. These bugs can cripple key functionality of the affected device, meaning that for what could be weeks at a time the devices are vulnerable to attacks from anyone who can send the device an iMessage, or even just sending a string of text to the device until Apple can find a fix and push it out to devices.

Apple has recently patched just shy of fifty security vulnerabilities that could have been exploited, spanning not just iOS, but also on macOS and WatchOS. The iOS 10.3.3 update that was pushed to supported iPhones, iPads and iPod Touch devices was noted as resolving 47 flaws, included in these flaws were remote execution code holes in what is known as the WebKit browser engine (Nichols, 2017). The WebKit browser engine is the basis of Apple's browser, Safari, and is essentially a rendering engine for a browser to use. Google used to use WebKit, but since 2013 it does not use WebKit as Google have created their own version which goes by the name of Blink (Finley, GOOGLE CHROME BREAKS UP WITH APPLE'S WEBKIT, 2013). The issues that are found and fixed can have their information found by searching their CVE code on the internet, which will have a user ending up on the Common Vulnerabilities and Exposures website. From here, users can find a more in depth information on the National Vulnerability Database (NVD) website. In September of 2015, Apple removed 40 applications from the App Store, due to them being infected with something called XcodeGhost. XcodeGhost was a form of malware which was designed to turn Apple devices into a large scale botnet. While Apple is more strict compared to Google with regards to the applications that can be found on their respective stores, this is an instance where Apple was not perfect with their applications, and malicious code got through with legitimate code, which was what made it harder to detect (Kaspersky Lab, 2015).

Of the flaws found, 23 were found in WebKit, with 16 of the issues being memory corruption errors. An example of what can be found is when one of the fixed memory corruption error codes is searched is the CVE site (CVE, 2017), which will give a description of the bug, and a link to the NVD website, which has a lot more information on the bugs. If exploited, these memory corruption bugs could allow remote code execution via malicious webpages (Nichols, 2017), which results in a form of a Denial of Service attack (CVE, 2017). Another security vulnerability that was addressed was that of a bug in Safari, in the Safari Printing section of the browser, which allowed an attacker to completely freeze a user's browser by flooding it with print dialogue boxes. Another security vulnerability that was addressed was one which was within the Messages application, which caused the application to be able to be attacked and would crash as a result (CVE-2017-7063). There was also remote code being able to execute in, or access restricted memory space, which was also fixed (Nichols, 2017). This remote code execution or code access to restricted memory space is a rather troublesome problem as there is usually a reason for the restriction on the memory, whether it be due to sensitive data being stored there or just private for the OS to use.

One bug that was of utmost importance to fix, that was fixed with the 10.3.3 patch was a problem with code that was within the Wi-Fi module in all iPhone devices from the iPhone 5 and up, iPad 4 and up, and on the iPod Touch 6 Gen. The bug went by the name "BroadPwn", as it was a problem with the Broadcom Wi-Fi chipset, and had the ability to be devastating

should it have been exploited. If it was not patched and was exploited, all an attacker had to do was get within Wi-Fi range of the target in order to be able to infect it, but the infected device could also be turned into a rogue access point, which could then infect nearby phones. A very large problem in this current day and age where many people leave their Wi-Fi on when going out of range of a connection, and are so dependent on the internet. This would have been the first Wi-Fi worm, security researcher Nitay Artenstein, the discoverer, said (Greenberg, 2017). This flaw also affected Android devices, as it was recently patched by Google as well, in their July security update (Spring, 2017). In an interview with Wired, Artenstein was quoted as saying that "As hackers search for increasingly rare attacks that don't require any interaction from users, like opening a malicious page in a browser, or clicking a link in a text message, they'll focus on third-party hardware components like Broadcom's chips" (Greenberg, 2017), which makes sense as he was also quoted as saying "…components sourced from third-party companies whose code Apple and Google don't entirely control.", which does also makes sense but it makes one wonder why if they are getting so many parts from companies from the likes of Broadcom that either Apple or Google aren't having someone or even a team of people going over the code in such vital parts in a modern day smartphone. Artenstein had wondered for years about Broadcom's Wi-Fi chip, and wondered if it may offer ways into the insides of a smartphone. The actual kernel of a smartphone has protection measures built in to prevent attackers from exploiting it, such as address space layout randomisation, and data execution prevention, which is designed to stop rogue malicious commands inside of data being computed and executed. The curiosity that Artenstein had stemmed from the fact that the Broadcom Wi-Fi controllers do not have these types of protection on board (Greenberg, 2017). This bug is kind of like having a default password that works for every computer log in everywhere; if it is found and isn't fixed, it can be exploited across many devices and cause vast amounts of damage. Artenstein was in the middle of reverse engineering the Broadcom code used on the Wi-Fi modules when he discovered that the source code had been leaked on popular code sharing site GitHub he began scouring through it for bugs and he began to find bugs throughout the code. He ended up finding the bug in question, BroadPwn (National Vunerlibility Database, 2017). The bug works by exploiting the handshake process for joining familiar Wi-Fi networks, where a piece of data that came from the Wi-Fi access point was not properly constricted and could cause a bug known as a "heap overflow". Those who know what they are doing would have been able to create an attack that could exploit this bug, and have an access point send data that could corrupt the Wi-Fi module's memory, which would in turn result in in an overflow into other parts of memory to run as commands. It is essentially creating a "write anywhere, brute force entry" kind of attack. Artenstein said that this kind of attack in other areas, such as the previously mentioned kernel would be very hard given its defences, but it works very well in memory like that set up in Broadcom's Wi-Fi modules. He says that "it's a pretty special bug" (Greenberg, 2017). The user may not even know that an attack has occurred, and that their Wi-Fi module was now being used as a rogue module spreading this bug around. Artenstein believes that the step from gaining control of the Wi-Fi module to getting into the kernel would not be hard for a motivated, well-resourced attacker (Greenberg, 2017).

There will most likely be more problems like this which currently remain undiscovered, but they will be very hard to find for the vast majority of people so they most likely will not be exploited. As stated previously, the companies that depend on third party companies for parts such as Wi-Fi controllers for their devices will have to be more vigilant with regards to the strength of the code on these devices. This code was also not commonly available code, so that also decreases the odds of another large flaw like this just being found.

While this was more of a cross platform problem due to it being a hardware issue, iOS has had issues in the past with the lock screen being able to be "bypassed" in such a way that allows people to access the photos of the device, which usually is not available even if there is a camera shortcut to get quick access to the camera. To access the photo library on a device with a lock on the device usually requires the device to be unlocked to view or modify the photos stored on the device. While it can be done, it does take some time to do, and it takes a rather substantial amount of physical time with the device to be able to pull off and it is not a way of guaranteed access, with access being more luck than anything. The website Gizmodo has an article (Warren, 2016) talking about how to do it, with a link to a video showing how to actually get into a device using this method (iDeviceHelp, 2016). This was done on iOS version 10.2. To put it simply, the steps required of the attacker are as follows: Know the number of the device, which can be done by asking Siri, the iOS Virtual Personal Assistant. They then need to call the number from another device, and from the incoming call screen they will have to choose the "message" option. They will then have to choose the "custom" message, which is the message that would get sent to the calling device. When in the custom message, Voice Over needs to be enabled, which can be done by asking Siri. The attacker then needs to double tap on the recipient field of the message, while simultaneously tapping on a random key on the keyboard. This should then open up a field on the SMS which will allow the attacker to then choose who the message gets sent to. From this point, the attacker can see all of the contacts stored on the device. They can also create a new contact, which is what will allow the attacker to get access to the photo gallery. When they create a new contact, they can choose to either leave the contact image blank, take a picture for the image, or select from the photo gallery which will give them access to the devices photo gallery (Warren, 2016). The video stated that users should actually disable Siri from the lock screen, so unless this way of accessing part of the allegedly protected contents of the device has been patched with the latest iteration of iOS, users should disable Siri from the lock screen just to be safe (iDeviceHelp, 2016). The ability to be able to access protected data like this is not good. While this could be a somewhat easy fix, just by not allowing voiceover to be enabled by Siri from the lock screen as some people would use Siri for things on the lock screen, should they not wish to unlock their device to use it normally.

There is another case of using Siri to access parts of a user's device that should not be able to be accessed by an outsider. This is on iOS 10.3.2, and the 10.3.3 beta, demonstrated on YouTube in may 0f 2017, by user EverythingApplePro (EverythingApplePro, New Siri Lockscreen Bypass on iOS 10.3.3! Disable Cellular Data, Read Texts & More , 2017). It required that the attacker must wake the device and get Siri up on screen which is done by double tapping the home button, and all you have to do is ask Siri to read all of your messages. Siri will proceed to read all unread messages, even if the content cannot be seen on screen, or is hidden on screen. Given that the lock screen is meant to keep people from knowing private data on a device, being able to get the personal assistant of a smartphone to read out new text messages is somewhat of a privacy flaw. Siri can also post to your Facebook wall, see the most recent caller, and even send text messages (EverythingApplePro, New Siri Lockscreen Bypass on iOS 10.3.3! Disable Cellular Data, Read Texts & More , 2017). These types of things should not be able to be done from the lock screen. Cellular data can also be turned on by using a bit of a bypass method with Siri as well. All you have to do is when talking to Siri is say "cellular data", and the cellular data toggle switch will pop up on screen. This could end up costing user's money and data if left on, and it allows posting to Facebook by using the Siri method mentioned just before. Doing these things should be done after the lock screen has been unlocked, and not from the lock screen itself. These are also quite easy to do as by default Siri is on for the lock screen, so if someone didn't use Siri or know it was on the lock screen then

this could most likely be done to them with ease. (EverythingApplePro, New Siri Lockscreen Bypass on iOS 10.3.3! Disable Cellular Data, Read Texts & More , 2017).

It seems that Siri, the iOS personal assistant is a weakness in the security of the operating system. While Apple wish for Siri to be able to help you do anything and everything from anywhere on the device, the lock screen proves to be a tough one to get right. For instance, people will wish to have their text messages read out to them while driving. This is all well and good. They may wish to reply to the message using Siri to dictate. Also fine. This is a continent feature to have. It appears that it has become a security flaw, as Siri is able to be abused in such a way that it can allow unauthorized users access to potentially sensitive data stored on the device. Although this appears to be a security flaw, this does bring up an interesting talking point: Virtual Personal Assistants.

Virtual Personal Assistants

A Virtual Personal Assistant is a software based entity inside of a devices OS which is designed to be able to help the user in some way, whether it be with setting reminders, answering simple questions, or reading out emails and dictating a user's replies. These days, the audible response from personal assistants has become more fluid and natural which leads to a less artificial sounding response from the assistant. To some degree, conversations can be held, with the modern assistants being rather interactive. They can utilize what is known as Artificial Intelligence. These assistants are becoming more and more powerful, with the use of neural networks and machine learning being utilized in order to help the assistant learn, develop and adapt to its specific use case. Currently on mobile devices there are three major personal assistants. iOS has Siri, as talked about previously. Android has Google Now and Google Assistant, but there are others that are also on the Android platform, such as Samsung's Bixby. Windows Mobile has Cortana, which is the same assistant which can be found on Windows 10. All of these assistants can be set up to be voice activated, usually using either the phrase "hello Siri / Cortana", or in the Google Now case, "Ok Google".

Siri can also be found on other Apple products, such as the Apple TV, Apple Watch and as of the macOS Sierra update Siri can be found on Apple's laptops and desktops. They are really pushing their VPA all over their eco system. Siri is known for being able to actually have a conversation that can help the user decide on what to do next after being asked a question and replying with relevant answers or suggestions on what to do. Siri is able to understand human language at a deep level, much in thanks to its artificial intelligence algorithms. While it is good at interacting with native applications on the device, it cannot interact with third party applications (Das, 2017).

Google Now is a personal assistant that is able to be found on Android devices, but can also be used on iOS devices as Google Now is not platform specific like Siri. While being able to be run on iOS, it is limited in comparison to what it can do on Android. Google Now lets users quickly search the internet and perform tasks, as a VPA should. You can set alarms, adjust volumes of your device and even post to social media. Google Now can be set up to be able to be used from the lock screen, and is able to do most things it can do when unlocked when the device is locked, but this VPA does not get to know the user. This is when Google Assistant comes in to play. Google Assistant could end up completely replacing Google Now, and has replaced Now on Googles flagship Pixel devices. It performs the same tasks as Now,

but is more of an interactive assistant as it can be conversed with, albeit to a limited degree at this point in time (Purewal, 2016).

Cortana is also a cross device VPA, like Siri, as it can be found on both Windows 10 Mobile devices and on Windows 10 equipped computers. Cortana is designed a bit different to how Siri and Google Now / Assistant have been designed as Cortana is more focused on interaction with the users personal life, focusing on reminders and calendar appointment, but it can still be asked questions like the other two VPAs. It integrates well with other Windows applications, but it is also able to be used on iOS and Android equipped devices, as Google Now can run on not just Android equipped devices. While it is a cross platform VPA, it is less advanced compared to the likes of Apple and Google's offerings, and it is reported to be the least used feature on the Windows 10 operating system (Das, 2017).

While the VPA can bring some convenience to the end user, it can open up holes in the software surrounding it, or even itself. As seen previously, Siri can be exploited in ways which allows attackers to get to allegedly protected data. The security risks and potential threats that are in relation to different VPAs are still being ascertained, and magnitude quantified (Das, 2017).

The major issue is privacy. While the end user holds the majority of the conversation and the VPA merely provides responses from servers depending on what the VPA can do with the device, meaning that conversation threads or even just questions asked at the device, are sent back to the respective VPA creators (Das, 2017). This is why most VPA services require an active internet connection for some features. This can be looked at both positively and negatively. This can allow developers to see what users ask of their VPA, and to increase the strength of the overall VPA by targeting the most commonly requested actions of said VPA. This is good, but it still means that the data is able to be harvested easily. This will all be legal, as you agree to EULAs when using either the OS, downloading the VPA application or upon first boot of the VPA. The question of how secure is the data which gets sent back to the servers comes up, as privacy is a right and companies should do their best to protect your privacy. "Up to this point, there have been no known studies which have been conducted to examine the depth of Security of these particular lines of communication." (Das, 2017). There is a chance that these lines may be unencrypted, thereby making it easy for those who wish to listen in to listen in and see what is being send and received by the device's VPA. There is a chance that while you have a conversation with the likes of Siri or Google Now, you may be sending data to a completely different country, where the likes of wiretapping and eavesdropping is able to be done, with little to no protection for the end user (Das, 2017). Potentially a "man-in-the-middle" attack could occur, but this would be unlikely. The conversations that users have with their VPAs are stored, which could be for legal reasons, convenience for users, or for data analysis by developers. Apple's retention policy is stated as being 18 months, but the retention length for Microsoft and Google are not known.

For an assistant to be effective, they have to learn and adapt to end users. This means that developers need to know what to program for, and the only way for them to know what to do with the VPA is with data provided by end users.

Malware and Viruses

MacAfee is one of the world's leading cybersecurity companies. They provide security for both home usage and business usage. In their 2017 threat prediction, they predict that ransomware would continue to grow, just as it did in 2016. One of the bigger problems in 2016 was that of Android/Jisut. This was an attack in which the ransomware got into the device and changed the device's lock pin and demanded payment in the form of bitcoin or prepaid card. These payment methods were chosen as they are hard, if not impossible to track. This type of "lock out users from their device and charge bitcoins in return for the unlock code" ransomware was globally known during May of 2017 which targeted the Windows OS, which encrypted the storage media and then charged users bitcoins in order to unlock their device (BBC, 2017). There are some known ways to reset or clear a passcode on Android, but some come at a cost. To get your device back under their control, users could try reset their password with their Google account. If this doesn't work due to how the attack works, then users could try to use a desktop application by the name of Android Data Recovery (Sophia, 2015). There is a tool which will remove the lock screen protection method within a few minutes. Should these not work, then the data is will have to be wiped, but at least the device will still be able to be used. Users will have to factory reset their device, which will fix the password lock out issue but at a cost of the data that is stored on the device (Sophia, 2015).

With many potential bugs in major operating systems such as the likes of broadpwn in Android and iOS, how does one protect themselves? Users cannot protect themselves from bugs like broadpwn, it is down to OS developers to protect users against bugs and potential exploits like this. While they may not be the ones to discover the bug that could be exploited, it is down to them to patch it and push the update to all devices which need to be updated, assuming the device is supported. Users can however protect themselves from other attacks, such as malware attacks, most of which will be using general targeting code, nothing like what would have had to have been used in the likes of broadpwn, as it was a specific problem in a proprietary set of code which had to be exploited. Kaspersky Lab, a Russian multinational cybersecurity and antivirus firm have created a page on their USA website which outlines the risk factors, types of malware and how to protect yourself.

Kaspersky state that 87 percent of all Android smartphones are exposed to at least one critical vulnerability, according to the University of Cambridge (Thomas, Beresford, & Rice, 2015). A company by the name of Zimperium Labs discovered earlier in 2015 that 95 percent of Android devices could actually be hacked with a simple text message (Zimperium, 2015). This means that should you run an Android device, you would have been at risk of being attacked, damaged or maybe even hijacked. While this was an Android problem, previously mentioned in the iOS portion, iOS is susceptible to malicious code, as with the XcodeGhost issue that was talked about. There are various types of malware that a device can be infected with. They range from the likes of banking malware, mobile ransomware, mobile spyware, MMS malware, mobile adware and even SMS Trojans.

Banking malware is malware that is designed to target a specific user, the ones who rely more on their mobile device for banking related needs than their desktop computers. The Kaspersky page states that "more than 1.6 million malicious installation packages were found in Q3 of 2015, many of them Trojans designed to infiltrate devices and then deploy, collecting bank login and password details, which are then sent back to a command and control (C&C) server" (Kaspersky Lab, 2015). They also state that of the growth of all malware in Q3 of 2015, mobile banking Trojans were the fastest growing. In 2015 there was a banking Trojan that intercepted text messages which contained any information pertaining to finances, and then

sent a copy of the test message(s) via email back to the attacker, which in turn gave them the information that they could use to get into financial accounts (Brook, 2015).

Mobile ransomware has been mentioned above, so there is no need to re go over it, but Kaspersky did state that "malware creators leveraged both improved smartphone performance and the anonymous Tor network to infect devices and encrypt stored data" (Kaspersky Lab, 2015). This means that it is harder, if not impossible to find the source, and since as mentioned before the ransom is usually bitcoin which cannot be traced, there is a very slim chance of ever finding the attackers. For reference, "Tor" is The Onion Router, which was created by the US government who desired privacy and anonymity when communicating (Scharr, 2013). The idea of Tor was to route users through many nodes, like layers in an onion, hence the name. Due to it being only government officials who had the use of the network it would have been a lot easier to track who was doing what on the network. This is when they released it to the public in the late 90s, so there were more nodes to route users through (increasing speed), and more users who use the service means that it is harder to know who is who. With Tor, users can access the normal web, the deep web, which is all the unindexed sites or pages on the internet but you do need a link to get to them, and the dark net, which is where drugs can be purchased, guns can be purchased, and much more, with any transactions usually being bitcoin due to its ability to not be tracked. Mobile spyware is generally loaded onto the device as a program which monitors activity, records location and pulls vital information, with the likes being usernames and passwords for the likes of email or banking sites (Kaspersky Lab, 2015).

Spyware is usually packed with applications that users would not think of spyware being included in, and while the application lays dormant, the spyware is fishing for the vital data. The only way users may notice spyware on their device is if they have performance slowdowns due to the background spyware task hogging resources or if the user runs a sweep for viruses and malware (Kaspersky Lab, 2015). While downloading an application that could potentially harbor spyware is risky, it is the users choice to download the application. Receiving MMS malware is not a choice. CSO Online noted that a vulnerability in Android's media library Stagefright made it possible to attacker to send a text message that was embedded with malware to any number (Korolov, 2015). This means that if a company that was hacked had a list of users' phone numbers stored away somewhere, these users could be vulnerable if those who used MMS malware attacks used the hacked and stolen list of device numbers. Users don't even have to acknowledge the message for the malware to deploy. This means that attackers can still get root access to a user's device. This problem was quickly fixed as it was a major flaw that Google could not sit on for long, but it did still prove that text based infections could exist and could do damage (Korolov, 2015).

Mobile adware is another problem that devices can suffer from. Adware makers generally depend on the clicks and downloads that they receive, and ZDNet was stated as saying that there is a type of code called "malvertising" code, which infects and roots the device, which then forces it to download specific adware types and allows attackers to steal personal information (Osborne, 2015). The ZDNet article also stated that the auto-root applications that get installed via malicious mobile campaigns are rather worrying within the Android ecosystem. The article goes on to say that adware is becoming trojanised, where the adware comes in via legitimate applications which allows malicious code to be installed at the same time as application install, and steals users' data, all after rooting the device to firmly stick itself inside the device (Osborne, 2015).

The last major type of malware is that is rampant enough to be problematic is that of SMS Trojans. These prey on one of the most common feature of not just a smartphone, but really any phone that can be purchased new these days; text messaging. SMS Trojans send messages out to "premium rate" numbers around the world, which in turn stacks up on the users phone bill (Kaspersky Lab, 2015).

Given there are many different attacks around the world that can affect a user at any time, how best can they defend themselves against such attacks? Kaspersky also provides tips for users on how to defend themselves. Use secure Wi-Fi networks. Secure Wi-Fi networks are good at stopping attackers snooping in on data packets sent in between a users' device(s) and the router, and this also stops them from being able to carry out man-in-the-middle attacks. A man-in-the-middle attack is where an attacker gets into a conversation that is being carried out by two parties, they then impersonate both parties and can from there gain access to information that each party is trying to send and receive. This attack allows the attacker to send and receive data that is meant for someone else, or not meant to be sent at all. No party will know what has happened usually until it is too late (Veracode, n.d.). The next tip Kaspersky give for users to do is to watch their email. They say that many attackers still rely on malicious email attachments to infect the device that they are opened on. They say to not click on any links in emails or other messages, as these links may direct users to phishing or malware sites, and that this applies to not just mobile platforms but all platforms (Kaspersky Lab, 2015). Phishing scams are scams where attackers try and steal money through getting the users details, whether it be over the phone, getting them to fill out an online form or even infecting their computer with malicious software to steal the information that they want. Usually these emails include suspicious links, poor grammar, and even threats to try and get users to give up the desired information (Microsoft, n.d.). Kaspersky also tell users to only download applications from trusted sources, and they they are legitimate applications, not "havens for mobile malware" (Kaspersky Lab, 2015). The final two suggestions for users to do in order to keep safe is to install some form of antivirus software, and to not jailbreak (iOS) or root (Android) their device. Antivirus software can scan the users entire device to see if there are any applications or downloaded malicious code on the device, and by not rooting or jailbreaking their device users decrease their risk of infection from third party sources. "Stay unrooted and benefit from automatic security updates and patches" (Kaspersky Lab, 2015).

Jailbreaking

Jailbreaking and rooting devices poses an interesting question, why should users need to do it? What features do mobile operating systems in 2017 not offer users that most likely would have already been implemented should enough people want them? Originally, jailbreaking was for users who wished to use an iPhone off of AT&T's network, a telecommunications provider in America (Dunford, 2017). Jailbreaking brought about features that are now commonplace on iOS, such as changing ringtones (Dunford, 2017). Installing applications that aren't found on the App Store or in Googles case the Play Store, is referred to as "side loading". After the release of the "jailbroken" app store by the name of Cydia, iOS jailbreaking became very powerful. There is two types of jailbreak; tethered and untethered. Tethered jailbreaking is where the Apple device needs to be plugged into a computer and booted up with special software on the computer, or else the Apple device will not be in its jailbroken state (iPhoneFAQ, n.d.). For untethered jailbreaks, everything that is required to get the device into its jailbroken state when rebooted is contained on the iPhone itself (iPhoneFAQ,

n.d.). Cydia allowed iPhone users with jailbroken devices to install applications and tweaks that were not available at the time. The creator of Cydia claimed back in 2008 that Cydia was installed on 10 percent of all iPhones at the time (Dunford, 2017). While jailbreaking allowed install of other applications and features, it means that anything that users installed was third party. They weren't developed by Apple or approved to be in Apple's App Store, so there is a higher chance compared to apps from the App Store that they could be containing malicious code, and given the higher permissions that the jailbreak allows, they could potentially access more data than malicious code that would be attacking a device that was not jailbroken. Developers have a reason for not allowing some applications on the App Store, whether it be the feature will come out in a future iteration of iOS, or maybe it is a security issue. It could be that the permissions required for the feature to work is too high for what application developers are allowed access to, and should be implemented by the OS developer as they can monitor exactly what this feature can access and what it should not access, which may not be top priority for some application developers. The OS developer would also be able to ensure stability across many device configurations and OS implementations. Jailbreaking also meant that updates for the device would break the jailbreak, meaning any features or tweaks installed by the user would not work until jailbreaking groups had jailbroken the OS for those who wish to use jailbreaks. If they chose to not update their OS, they would could be vulnerable to attacks, or would eventually become vulnerable to attacks as their software was old and out of date. Now days jailbreaking is not as popular as it once was as, mainly due to Apple implementing features that were once only available via jailbreaking. These groups or singular people who were once jailbreakers are now "seen as valuable security analysts" (Dunford, 2017). Many of the best jailbreaking minds are now employed in the tech sector. If they aren't employed by companies like Apple, they may work freelance, meaning that sell exploits to security firms for large sums of money. Given the little amount of devices that iOS has to work on, it will be largely secure. Being so secure, any flaws would be worth a lot, should they be uncovered (Dunford, 2017).

Rooting

Rooting a device is the Android equivalent of jailbreaking a device. However, with rooting Android devices, there is no tethered or untethered root, there is only an "untethered" root. Usually when a device is rooted, it is with the intention of installing or "flashing" a custom ROM, which is a modified version of Android OS. With Android devices generally packing more raw power than iOS-running smartphones, there is more hardware control when rooting, as compared to that of jailbreaking. Users can actually overclock and over-volt their devices' CPU, leading to performance boosts. While doing this leads to better performance, battery life will most likely suffer, and pushing a device's hardware too far can lead to unrepairable damage. Rooting is also used for getting rid of carrier bloatware, which telecommunications carriers load onto devices that are sold by them, as compared to an "international" device, which has just whatever software the manufacturer intended it to have on it. While it may be good to have a faster smartphone with more storage, it can come at a cost. Rooting a device can have many repercussions. These can range from voiding the devices' warranty, bricking the phone, exposing security risks and actually losing some application support (Digital_Trends_Staff, 2017). The warranty of the device will be void as the user is modifying the device beyond how the manufacturer intends it to be modified, thereby relieving them of having to stand by the device should there be problems with the OS, or the hardware itself. When rooting a device goes bad, either in the rooting phase or installing a custom ROM, a device can become bricked. Bricking a device means that a users' devices' memory has become

corrupted. One of the bigger problems with rooting was mentioned in the list before, with the problem being security. Android is already fragmented as it is, rooting a device means that it is not able to get official Android updates that get pushed by the manufacturer or Google, depending on the device. Because rooting a device allows access beyond some default Android security features depending on whether you root and keep the stock OS or flash another one, this means that anything that attacks the rooted device will have an easier time getting the desired results from the attack as some Android security features that the attacker may have already been bypassed by the root access (BullGuard, n.d.).

With more and more features becoming default to mobile operating systems or becoming applications in respective stores, the incentive to root or jailbreak devices seems less enticing compared to when these operating systems were in their infancy and attackers weren't yet targeting mobile devices as much as they now are. If you are to root or jailbreak you lose software updates unless users go back to what OS and firmware the device shipped with (Android), un-jailbreak the device, or restore the device to get the factory OS back again (Apple). The security flaws definitely do not seem worth it, as they introduce more flaws by circumventing some default OS security features, which can allow side loading of applications that aren't on default application stores for a reason, and various other security tweaks which can make it easier for attackers to attack and get desired results.

Conclusion

Mobile device security in 2017 seems to be in a good place for the majority of the population, with operating system manufactures keeping older devices still protected from large bugs like that of broadpwn, and general security tweaks, or allowing older devices to still have the latest operating systems in order to keep them secure and somewhat protected. Smartphones are offering more and more protection features for devices, such as pattern locks, passwords, and with the likes of fingerprint recognition and facial recognition systems improving from where they first started, now becoming harder to trick to gain access to the device. With Apple's iPhone X relying heavily on facial recognition for device protection and access, it will be interesting to see if such a bold step towards a newer technology for mobile devices will be a strong enough protection feature to be able to only have it as the main protection feature for the device. On the respective application stores, there are many antivirus applications that can scan your device to check and protect users from malware and viruses, so they won't have to worry as much. While users wouldn't have to worry as much if they have protection like antivirus applications and default operating systems protections, it does not mean that users can do whatever they wish on their devices, they still need to be careful as no operating system is completely secure. Users should uninstall applications that they are no longer using in order to make sure that they aren't risking exposing themselves to more than they need to. But as everyone should know, the most secure smartphone is no smartphone at all.

[WC: 15901]

Bibliography

Alciné, J. (2015, June 29). *Google Photos, y'all fucked up. My friend's not a gorilla*. Retrieved September 14, 2017, from Twitter: https://mobile.twitter.com/jackyalcine/status/615329515909156865

Android Authority. (2016, July 9). *How fingerprint scanners work: optical, capacitive, and ultrasonic variants explained*. Retrieved July 22, 2017, from Android Authority: http://www.androidauthority.com/how-fingerprint-scanners-work-670934/

Arici, A. (2017, April 4). *How does the Samsung Galaxy S8 iris scanner work?* Retrieved August 2, 2017, from Android Guys: http://www.androidguys.com/2017/04/04/how-does-the-samsung-galaxy-s8-iris-scanner-work/

Bates, P. (2016, June 9). *What Is The Most Secure Mobile Operating System?* Retrieved September 2, 2017, from Makes Use Of: http://www.makeuseof.com/tag/secure-mobile-operating-system/

BBC. (2017, May 13). *Cyber-attack: Europol says it was unprecedented in scale*. Retrieved September 5, 2017, from BBC: http://www.bbc.com/news/world-europe-39907965

Brook, C. (2015, February 3). *NEW BANKING TROJAN TARGETS ANDROID, STEALS SMS*. Retrieved September 13, 2017, from Kaspersky Lab: https://threatpost.com/new-banking-trojan-targets-android-steals-sms/110819/

BullGuard. (n.d.). *The risks of rooting your Android phone – BullGuard* . Retrieved September 20, 2017, from BullGuard: http://www.bullguard.com/bullguard-security-center/mobile-security/mobile-threats/android-rooting-risks.aspx

Chacos, B. (2014, April 4). *Microsoft's universal Windows apps run on tablets, phones, Xbox, and PCs*. Retrieved September 1, 2017, from PCWorld: https://www.pcworld.com/article/2138625/microsoft-introduces-universal-apps-that-scale-across-phones-tablets-and-pcs.html

Claburn, T. (2017, August 17). *What code is running on Apple's Secure Enclave security chip? Now we have a decryption key...* Retrieved October 10, 2017, from The Register: https://www.theregister.co.uk/2017/08/17/apple_secure_enclave_decrypted/

CVE. (2017, July 19). *CVE-2017-7055*. Retrieved August 13, 2017, from Common Vulnerabilities and Exposures: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-7055

Das, R. (2017, June 1). *The Virtual Personal Assistant and Its Security Issues*. Retrieved August 29, 2017, from Infosec Institute: http://resources.infosecinstitute.com/virtual-personal-assistant-security-issues/

Digital_Trends_Staff. (2017, April 19). *HOW TO ROOT YOUR ANDROID PHONE OR TABLET IN 2017 (AND UNROOT IT)*. Retrieved September 19, 2017, from Digital Trends: https://www.digitaltrends.com/mobile/how-to-root-android/

Duke University. (n.d.). *LZW Data Compression*. Retrieved October 10, 2017, from Duke University: https://www.cs.duke.edu/csed/curious/compression/lzw.html

Dunford, D. (2017, July 18). *iOS Jailbreaking Is Coming To An End*. Retrieved September 19, 2017, from Value Walk: https://www.valuewalk.com/2017/07/iphone-jailbreak-8-ios-jailbreaking-11/

Eadicicco, L. (2017, January 19). *Watch Out For This iPhone-Crashing Text Message*. Retrieved August 8, 2017, from Time: http://time.com/4637574/iphone-crash-text-2017/

EverythingApplePro. (2017, May 27). *New Siri Lockscreen Bypass on iOS 10.3.3! Disable Cellular Data, Read Texts & More* . Retrieved August 22, 2017, from YouTube: https://www.youtube.com/watch?time_continue=1&v=_Zy_HzK_G4o

EverythingApplePro. (2017, January 17). *This Text Will CRASHANY iPhone* . Retrieved August 8, 2017, from YouTube: https://www.youtube.com/watch?v=G0iPhSuiMpk

FIDO. (n.d.). *FIDO Home Page*. Retrieved October 10, 2017, from fido alliance: https://fidoalliance.org/

Fink, E. (2014, May 22). *I Made Obama's BlackBerry*. Retrieved October 10, 2017, from CNN: http://money.cnn.com/2014/05/22/technology/security/nsa-obama-blackberry/

Finley, K. (2013, March 4). *GOOGLE CHROME BREAKS UP WITH APPLE'S WEBKIT*. Retrieved August 11, 2017, from Wired: https://www.wired.com/2013/04/blink/

Finley, K. (2017, September 13). *Can Apple's iPhone X beat Facial Recognition's Bias Problem*. Retrieved September 14, 2017, from Wired: https://www.wired.com/story/can-apples-iphone-x-beat-facial-recognitions-bias-problem/

GdD. (2013, October 14). *How secure is Android's facial recognition?* Retrieved September 8, 2017, from Stack Exchange: https://security.stackexchange.com/questions/43808/how-secure-is-androids-facial-recognition)

Google. (n.d.). *Position Sensors*. Retrieved July 21, 2017, from Android Developers: https://developer.android.com/guide/topics/sensors/sensors_position.html

Greenberg, A. (2017, July 27). *HOW A BUG IN AN OBSCURE CHIP EXPOSED A BILLION SMARTPHONES TO HACKERS*. Retrieved August 15 , 2017, from Wired: https://www.wired.com/story/broadpwn-wi-fi-vulnerability-ios-android/

Grenoble, R. (2015, May 27). *This Single Text Message Can Crash Your iPhone*. Retrieved August 6, 2017, from Huffington Post: http://www.huffingtonpost.com/2015/05/27/text-message-crash-iphone-_n_7452324.html

GSM Technical Specifications. (1996, July). Retrieved August 7, 2017, from etsi: http://www.etsi.org/deliver/etsi_gts/03/0340/05.03.00_60/gsmts_0340v050300p.pdf

GSMA_Intelligence. (2017, October 8). *GSMA Intelligence Home Page*. Retrieved October 8, 2017, from gsmaintelligence: https://www.gsmaintelligence.com/

GSMArena. (2017, July 25). *Temperature_Sensors_on_Smartphones*. Retrieved July 25, 2017, from GSMArena: http://www.gsmarena.com/results.php3?chkTemperature=selected

Hein, B. (2016, December 29). *New iMessage hack lets you crash friends iPhones via text message*. Retrieved August 7, 2017, from Cult of Mac: https://www.cultofmac.com/460230/new-imessage-hack-lets-crash-friends-iphones-via-text-message/

iDeviceHelp. (2016, November 15). *How to Unlock ANY iPhone Without Passcode Access Photos, Contacts & More iOS 9 /10 - 10.2*. Retrieved August 19, 2017, from YouTube: https://www.youtube.com/watch?v=LWJG5I8xCDU&feature=youtu.be

IEEE Standards Association. (2010). *802.15.1-2005 - IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN)*. Retrieved September 4, 2017, from IEEE Standards Association: https://standards.ieee.org/findstds/standard/802.15.1-2005.html

IEEE Standards Association. (2016). *802.11-2016 - IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. Retrieved September 4, 2017, from IEEE Standards Association: https://standards.ieee.org/findstds/standard/802.11-2016.html

iiiweb.net. (n.d.). *Cell Phone Tower Triangulation*. Retrieved August 3, 2017, from International Investigators Incorporated: https://www.iiiweb.net/forensic-services/cell-phone-tower-triangulation/

iPhoneFAQ. (n.d.). *What is tethered vs. untethered iPhone jailbreaking?* Retrieved September 18, 2017, from iPhoneFAQ: http://www.iphonefaq.org/archives/971144

Kaspersky Lab. (2015). *Mobile Malware*. Retrieved September 10, 2017, from Kaspersky: https://usa.kaspersky.com/resource-center/threats/mobile

Khanse, A. (2017, February 14). *Sandbox Sandboxing Software*. Retrieved September 2, 2017, from The Windows Club: http://www.thewindowsclub.com/sandbox-sandboxing-software

Korolov, M. (2015, July 27). *Stagefright vulnerability allows criminals to send malware by text*. Retrieved September 13, 2017, from CSO: https://www.csoonline.com/article/2952741/mobile-security/stagefright-vulnerability-allows-criminals-to-send-malware-by-text.html

Levi, J. (2011, October 9). *What Can You Do With A Barometer On A Smartphone?* Retrieved July 25, 2017, from PocketNow: http://pocketnow.com/2011/10/19/what-can-you-do-with-a-barometer-on-a-smartphone

Liu, J. (2016, March 28). *How are fingerprints stored in phones that use fingerprint verification?* Retrieved October 10, 2017, from Quora: https://www.quora.com/How-are-fingerprints-stored-in-phones-that-use-fingerprint-verification

Manea, A. (2014, August 26). *What Makes BlackBerry So Secure? Let's Look at Five Fundamentals of Security*. Retrieved October 10, 2017, from Blackberry Business Blog: http://bizblog.blackberry.com/2014/08/fundamentals-of-security/

Microsoft. (n.d.). *How to recognise phishing email messages, links or phone calls*. Retrieved September 14, 2017, from Microsoft: https://www.microsoft.com/en-us/safety/online-privacy/phishing-symptoms.aspx

Mishra, D. K. (2017, March 4). *How many different sensors are available in a smartphone?* Retrieved July 18, 2017, from Quora: https://www.quora.com/how-many-different-sensors-are-available-inside-a-smartphone

Moscaritolo, A. (2017, February 11). *President Trumps Mobile Phone: 5 Fast Facts You Need To Know*. Retrieved October 10, 2017, from Heavy: http://heavy.com/news/2017/02/what-kind-of-cell-phone-does-president-donald-trump-use/

National Vunerlibility Database. (2017, April 6). *CVE-2017-9417 Detail*. Retrieved August 17, 2017, from National Vunerlibility Database: https://nvd.nist.gov/vuln/detail/CVE-2017-9417

Nelenkov. (2015). *Password Storage in Android M*. Retrieved October 10, 2017, from Nelenkov.blogspot: https://nelenkov.blogspot.co.nz/2015/06/password-storage-in-android-m.html

NetMarketShare. (2017, July 17). *Mobile Operating System Market Share*. Retrieved July 17, 2017, from netmarketshare.com: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1

Nichols, S. (2017, July 19). *Apple hurls out patches for dozens of security holes in iOS, macOS*. Retrieved August 11, 2017, from The Register: https://www.theregister.co.uk/2017/07/19/apple_patches_ios_os_x_flaws/

Osborne, C. (2015, November 4). *Mobile malware evolves: Adware now breaks and roots your phone*. Retrieved September 14, 2017, from ZDNet: http://www.zdnet.com/article/mobile-malware-evolves-adware-now-breaks-and-roots-your-phone/

Purewal, S. J. (2016, October 4). *The difference between Google Now and Google Assistant*. Retrieved August 30, 2017, from Cnet: https://www.cnet.com/how-to/the-difference-between-google-now-and-google-assistant/

Qualcomm. (n.d.). *Snapdragon Security Platform*. Retrieved October 10, 2017, from Qualcomm: https://www.qualcomm.com/products/features/security

Rapid Web Services. (2017, August 26). *Re-Hashed: The Difference Between SHA-1, SHA-2 and SHA-256 Hash Algorithms*. Retrieved September 22, 2017, from Rapid Web Services: https://www.thesslstore.com/blog/difference-sha-1-sha-2-sha-256-hash-algorithms/

RotoView. (n.d.). *Magnetometer in Smartphones and Tablets*. Retrieved July 24, 2017, from RotoView: http://www.rotoview.com/magnetometer.htm

Rouse, M. (n.d.). *Dictionary Attack*. Retrieved October 10, 2017, from Search Secruity: http://searchsecurity.techtarget.com/definition/dictionary-attack

Samsung. (2017, July 17). *Samsung Galaxy S8 Specifications*. Retrieved July 17, 2017, from Samsung.com: http://www.samsung.com/global/galaxy/galaxy-s8/specs/

Samsung. (n.d.). *Heartrate Sensor*. Retrieved August 1, 2017, from Samsung: http://www.samsung.com/us/heartratesensor/

Scharr, J. (2013, October 13). *What is Tor? Answers to frequently asked questions*. Retrieved September 13, 2017, from Toms Guide: https://www.tomsguide.com/us/what-is-tor-faq,news-17754.html

SmartfoneArena. (2016). *Smartphone Sensors : Gyroscope, Accelerometer and Magnetometer : Basic Difference*. Retrieved July 20, 2017, from Smartfone Arena: http://smartfonearena.com/smartphone-sensors-gyroscope-accelerometer-magnetometer-basic-difference/

Spreitzenbarth Mobile Security and Forensics. (2012, February 28). *Cracking the Pattern Lock on Android*. Retrieved August 19, 2017, from Spreitzenbarth Mobile Security and Forensics: https://forensics.spreitzenbarth.de/2012/02/28/cracking-the-pattern-lock-on-android/

Spring, T. (2017, July 6). *GOOGLE PATCHES CRITICAL 'BROADPWN' BUG IN JULY SECURITY UPDATE*. Retrieved August 16, 2017, from Threatpost: https://threatpost.com/google-patches-critical-broadpwn-bug-in-july-security-update/126688/

T, N. (2014, March 6). *Did you know that "silent" text messages can be used to track your whereabouts?* Retrieved August 4, 2017, from PhoneArena: https://www.phonearena.com/news/Did-you-know-that-silent-text-messages-can-be-used-to-track-your-whereabouts_id53557

T, N. (2014, March 06). *Did you know that "silent" text messages can be used to track your whereabouts?* Retrieved October 10, 2017, from Phone Arena: https://www.phonearena.com/news/Did-you-know-that-silent-text-messages-can-be-used-to-track-your-whereabouts_id53557

Tested. (2014, December 29). *Testing Apple's Touch ID with Fake Fingerprints*. Retrieved July 23, 2017, from https://www.youtube.com/watch?v=2u4ZLGsw1zo

Thomas, D. R., Beresford, A. R., & Rice, A. (2015, October 12). *Security Metrics for the Android Ecosystem*. Retrieved September 11, 2017, from University of Cambridge: https://www.cl.cam.ac.uk/~drt24/papers/spsm-scoring.pdf

Tltcomb, J. (2017, January 18). *iPhone emoji message prank crashes phones with a single text*. Retrieved August 9, 2017, from Telegraph: http://www.telegraph.co.uk/technology/2017/01/18/iphone-message-prank-crashes-phones-single-text/

Triggs, R. (2017, September 15). *Facial Recognition Technology Explained*. Retrieved September 16, 2017, from Android Authority: http://www.androidauthority.com/facial-recognition-technology-explained-800421/

University of Pennsylvania. (n.d.). *Smudge Attacks on Smartphone Touch Screens.* Retrieved October 10, 2017, from Usenix: https://www.usenix.org/legacy/events/woot10/tech/full_papers/Aviv.pdf

Veracode. (n.d.). *Man In The Middle (MITM) Attack*. Retrieved September 14, 2017, from Veracode: https://www.veracode.com/security/man-middle-attack

Warren, C. (2016, November 19). *This Trick Apparently Lets You Bypass Any iPhone's Lock Screen*. Retrieved August 19, 2017, from Gizmodo: https://www.gizmodo.com.au/2016/11/this-trick-apparently-lets-you-bypass-any-iphones-lock-screen/

Weinberger, M. (2015, August 14). *There's one glaring flaw in Microsoft's Windows 10 strategy*. Retrieved September 2, 2017, from Business Insider Australia: https://www.businessinsider.com.au/microsoft-universal-windows-app-in-windows-10-are-deeply-flawed-2015-8?r=US&IR=T

Zimperium. (2015, July 21). *The Biggest Splash at BlackHat and DEFCON 2015*. Retrieved September 11, 2017, from Zimperium Blog: http://blog.zimperium.com/the-biggest-splash-at-blackhat-and-defcon-2015/