Vulnerabilities in Messaging Apps to Quantum Attacks

Samuel Clark
scsc2@kent.ac.uk

School of Computing
University of Kent
United Kingdom

Word Count: 7216

May 4, 2020

Contents

1 Introduction **2**

2 Cryptography in messaging apps **2**
2.1 RSA . . . 2
2.2 Diffie-Hellman . . . 3
2.3 Digital Signature Algorithm . . . 3
2.4 AES . . . 4
2.5 Elliptic Curve . . . 4
2.6 HMAC . . . 5

3 Quantum attacks **5**
3.1 Shor's Algorithm . . . 6
3.2 Grover's Algorithm . . . 7

4 Consequences of an attack **7**
4.1 Key Exchange . . . 8
4.2 Message Encryption . . . 8
4.3 Authentication . . . 8
4.4 Hash . . . 8

5 In depth analysis **9**
5.1 Overview . . . 9
5.2 WhatsApp . . . 9
5.3 Facebook Messenger . . . 10
5.4 iMessage . . . 11
5.5 Skype . . . 12
5.6 Viber . . . 13
5.7 Telegram . . . 14
5.8 Signal . . . 15
5.9 Wickr . . . 16
5.10 BBM Enterprise . . . 17
5.11 Wire . . . 17

6 Conclusion **18**

1 Introduction

In the current day and age, messaging apps are one of the most frequently used applications for communication. One of the most popular messaging apps used in the world, WhatsApp, has reported to have over two billion users [1]. Due to its popularity, it is a target for cyber attacks which have happened in the past [2].

It is common for messaging apps to use cryptographic primitives such as RSA, AES and Elliptical Curve (EC) cryptography. They are considered to be very secure with correct implementation however can be vulnerable to quantum attacks. Quantum computers are getting progressively better and algorithms which make use of the properties of quantum computers have been established which weaken the security of implemented cryptographic primitives [3]. Shor's algorithm and Grover's algorithm are two algorithms that make use of the quantum mechanical properties of quantum computer, the result of which can lead to vulnerabilities on these ciphers. The current bottleneck in quantum computers is with the limited number of qbits meaning that they are unable to implement and run these algorithms efficiently. In regards to Moore's law, it is only a matter of time before there is a quantum computer with enough qbits to implement these algorithms.

In this paper we will look at ten popular messaging apps and analyse the cryptographic primitives used within those to determine if they could be vulnerable to attacks. We will focus mostly on public key cryptography which is most vulnerable to quantum attacks.

2 Cryptography in messaging apps

Messaging app use different sorts of cryptographic technologies to ensure a secure connection. The systems which cryptography is needed within the apps includes hashing, message encryption, signing/authentication, key exchange and key generation. The most common technologies found in messaging apps included RSA, DH, DSA, ECC and AES.

2.1 RSA

Rivest-Shamir-Adleman (RSA) is a public key cryptosystem which is used most frequently for signing/authentication. RSA involves the use of a public and private key which can be in the following ways. The private key can be used to sign a message so that anyone with the public key can use that to ensure the message was from the intended sender. Alternatively, the public key can be used to encrypt a message that only the person with the private key can decrypt. This is a less common use of RSA due to it being

a relatively slow algorithm. Instead it can be used to encrypt a shared symmetric key to use for encrypting messages.

The security of RSA depends on the difficulty of factorising large numbers. The RSA algorithm uses two large prime numbers, p and q, to generate a large factor named n, which makes it difficult for classical computers to break n into its two factors[4]. Current attacks on RSA using classical computers have only been able to run at non-polynomial time. The best know attack uses the number field sieve algorithm which has been able to break RSA-795 (795 bits or 240 decimal digits) using CADO-NFS, an open source software. It took roughly 900 core-years to factorise a 795 bit RSA number[5].

2.2 Diffie-Hellman

Diffie-Hellman (DH) is an asymmetric key exchange protocol. Its primary use in messaging apps is to exchange a secret key between two parties.

Diffie-Hellman works with the use of modular exponentiation and works on the fact that $(g^a)^b = (g^b)^a$. Two parties, Alice and Bob, share two numbers in public, g and p, they then generate their own private number ,a and b respectively. Alice then computes g^a and sends it to Bob while bob computes g^b and sends it to Alice. From this they can then both compute the final number $(g^a)^b$ for Alice and $(g^b)^a$ for bob[6]. The security of DH is dependent on the ability to solve the discrete logarithm problem. Given an integer g, a prime number p and $g^{ab} \, mod \, p$ it is computationally difficult to find ab with the use of classical algorithms. There is not yet a classical algorithm which can solve the discrete logarithm problem in polynomial time.

2.3 Digital Signature Algorithm

The Digital Signature Algorithm (DSA) is a public key algorithm which is only used for digital signatures. A Federal Information Processing Standard (FIPS), set by the National Institute of Standards and Technology (NIST) proposed the use of DSA as apart of the Digital Signature Standard (DSS). DSA would then be used as the algorithm for the DSS to create a digital signature that can be used in Federal applications[7]. With the use of this algorithm, a private key is used to create a digital signature for a message. A public key is then used to verify the integrity of the message and ensure it was from the intended sender.

The algorithm involves four different operations which include generating a key pair, key pair management/distribution, Modular exponentiation is used within these operations and therefore DSA falls under the category of discrete logarithm cryptography. Similarly to DH, this means that the security of the algorithm is dependent on the ability to solve the discrete

logarithm problem.

2.4 AES

The Advanced Encryption Standard is a symmetric 128bit block cypher with varying key of 128, 192 and 256 bits, that is used to encrypt the plaintext within the messaging apps. It is considered to be one of the best encryption methods, considering speed and security. The operations of AES are performed on a 4x4 two-dimensional array of bytes. This is known as the state[8]. It makes use of a substitution-permutation network which takes 128 bits of plaintext and a key as an input. It then applies 10, 12 or 14 rounds for key lengths 128, 192 and 256 respectively of substitution and permutation to produce a 128 bit piece of cyphertext.
There are five operations which happen within AES which includes the following:
Key expansion - This is used before any rounds have taken place in which it generates a separate 128bit round key for each round plus an additional round key which is used at the start.
Add round key - This combines each byte within the state with a byte of the round key using an XOR operation. This operation happens before any of the rounds begin and at the end of each round.
Substitute bytes - Each byte in the state is substituted using a non-linear substitution table. This operation is performed in each round.
Shift rows - The last three rows in the state are cyclically shifted by varying offsets. This happens after the substitute bytes operation has been completed on every round.
Mix columns - Performs a linear transformation on four bytes of each column in the state. This operation occurs after the shift rows operation for ever round with the exclusion of the last round. Together with the shift rows operation this provides diffusion within the algorithm.
AES is considered a very secure cipher and is even integrated into some CPU's[9]. The best attacks on AES were performed on a variation of AES with 7, 8 and 9 rounds for 128,192 and 256 bit keys respectively[10].

2.5 Elliptic Curve

Elliptic Curve Cryptography (ECC) is a form of public key cryptography that makes use of elliptic curves. Variants of DH and DSA known as ECDH and ECDSA use ECC to allows for shorter key lengths to be used with an equivalent level of security. The security of ECC is dependent on the ability to solve the elliptic curve discrete logarithm problem which is similar to the discrete logarithm problem, only slightly more challenging allowing for shorter key length to be used. ECDH and ECDSA are considered much better to use when compared to their standard

counterparts and are therefore used more frequently than the standard variations. Different curves can be used which affect the security of the algorithm. Common curves used in messaging apps are P-521, which is a curve defined by NIST[11] and Curve25519, was designed for and is primarily used in with ECDH[12].

2.6 HMAC

Hash-based message authentication code (HMAC) is a method of authentication using hash functions. A MAC (message authentication code) is used for authentication and is used to verify the integrity of messages. HMAC is used to ensure that the message has not been altered in transit[13]. It is similar to a digital signature, like the ones produced by DSA, due to the fact that it provides integrity and authenticity of a given message. However, compared to digital signatures, HMAC use a symmetric key. Compared to a basic hash, it provides authenticity as well as integrity. The security of HMAC is dependent on the security of the hash function used and therefore varies.

3 Quantum attacks

Quantum computers open the world up to a whole new level of computational power. Some of the algorithms created to work with quantum computers are able to more easily solve mathematical problems, such as factorisation and discrete logarithms, when compared to classical computers. One of the main algorithms, Shor's Algorithm, has been shown to break the mathematical fundamentals behind DH and DSA. Another algorithm known as Grover's Algorithm can utilise quantum computers to severely reduce the security of algorithms such as AES in some cases.

Both of these algorithms make use of a quantum bit (qbit). A qbit is similar to a classical bit in the sense that it can be a 0 or 1 but it can be in a superposition of both those states simultaneously. This superposition property along with quantum interference is what makes these algorithms run more efficiently compared to on a classical machine. Two qbits provides an equivalent number of four classical bits. The number of qbits equal to classical bits is 2^N and N being the number of qbits. This means that the number of qbits to classical bits grows exponentially, allowing for polynomial computation of some algorithms.

Cryptographic Primitive	Risk
AES-128	Medium
AES-192, AES-256	Low
RSA	High
DSA	High
DH	High
ECC	High

3.1 Shor's Algorithm

Shor's algorithm named after the creator, Peter Shor[14], utilises the aspects of a quantum computer to be able to easily compute the factorisation of numbers in polynomial time. It can be implemented on classical machines but will run in non-polynomial time. It utilises a qbits quantum property of allowing it to be in a superposition of states.

Shor's algorithm works by taking the number that needs to be factored and using number theory changing into a different problem where instead you need to find the period of a long sequence of numbers. We can then calculate the period by using a quantum computer to have an input with a superposition of states which can find this period in polynomial time. To get the result we want quantum interference is used to destructively interfere to get a best answer for the period. This is done multiple times as the value returned is $\frac{x}{period}$ where x is an unknown number. By running this multiple times we are able to deduce the period and use that to find the factor of the original number.

Shor's algorithm is able to solve the factorisation of large numbers and the discrete logarithm problem. This means that cryptographic primitives that are dependent on the difficulty of these problems are no longer considered to be secure. It includes RSA, DH, DSA and ECC. All these technologies will no longer be considered to be secure when a quantum computer that is able to run Shor's algorithm is available. Extending the key length will also not make a difference as Shor's algorithm computes in polynomial time on a quantum computer. Therefore a large majority of the technologies used in messaging apps will no longer be secure which provide the underlying security for the app. We will see how each of the messaging apps are affected by Shor's algorithm later in the paper. The following table provides an overview of how the security of cryptographic primitives using factorisation and the discrete logarithm problem for their security compare to quantum security:

Primitive	Key Size	Classical Security	Quantum Security
RSA	1024, 2048, 3072	2^{80}, 2^{112}, 2^{128}	2^0, 2^0, 2^0
ECC	160, 256	2^{80},2^{128},	2^0, 2^0

3.2 Grover's Algorithm

For AES encryption, the best methods of attack are not practical on classical machines. Grover's algorithm, defined by Lov K. Grover[15], is a quantum algorithm that runs with a complexity of $O\left(\sqrt{N}\right)$. It is a brute force algorithm that can take search for an input to a black box function that produces a specified output. This algorithm can therefore have an impact on the security of AES.

Grover's algorithm works by using qbits in a superposition to speed up the search. As Grover's algorithm is a search algorithm the time for it to search N numbers on a classical machine is $O(N)$ however using qbits we get $O\left(\sqrt{N}\right)$. This can be derived by the previously mentioned property that N quantum bits is equal to 2^N classical bits. This therefore speeds up the search by a considerable amount.

This reduces the security of AES by only a small factor and not in polynomial time. This means that with larger key sizes AES is protected against Grover's algorithm. For AES-128 its security is reduced enough to be considered as weaker in terms of security and therefore is able to be broken by a quantum computer. This can be resolved by using AES-192 or AES-256 which have large enough key sizes to make an attack using Grover's algorithm unfeasible. The following table shows the impact of Grover's algorithm for all key sizes of AES:

Key Size	Classical Security	Quantum Security
128	2^{128}	2^{64}
192	2^{192}	2^{98}
256	2^{256}	2^{128}

4 Consequences of an attack

The consequences of an attack, whether it be a classical or quantum attack can lead to seemingly secure messages being intercepted by an attacker. An attack on communications can have a major impact on business and general users of the applications. Not being able to communicate securely is a major issue and can have serious consequences. For each of the technologies used in messaging apps to keep it secure, they can have different impacts.

4.1 Key Exchange

Key exchanges are the initial conversation between two users to establish a key to communicate securely. If an attacker was able to break the security of the key exchange, the initial key generated in the exchange would be compromised. From there the remaining communication between two users is compromised any additional keys used for message encryption can be seen to the attacker. The attacker won't need to break the message encryption as they will be able to obtain the keys generated for the encryption. The security used for the message encryption is irrelevant if the attacker has the key used in the encryption.

4.2 Message Encryption

If an attacker is able to only break the security used for the message encryption, they will be able to intercept and decrypt the messages between two users. This will have a major impact on users who will no longer be able to communicate privately. Communication on messaging apps are commonly used for business and confidential information. The encryption of messages is a vital part of the security of an app and without it, it leaves conversations open for an attacker to read.

4.3 Authentication

The security of authentication is very important in a messaging app. The use of a compromised authentication scheme can lead to an attacker having the ability to forge messages with the signature of another person. This will lead to messages being received with an uncertainty of who it was from. For important conversations, this can have major consequences. Authentication is a critical part of security and without it the integrity of the message sent is lost.

4.4 Hash

The hash of a message is important for ensuring a message hasn't been altered in transit. Messages that are sent may also include a payload such as a file or image, hashes are used to ensure the file is the intended file. An attack on the hash of a message could destroy the integrity of a message. If a message is altered and the hash of the message is compromised, a user will not be able to tell that the message has been altered. Hashes are also used to verify updates and security patches. An attack on the hash updates or security patches can lead to a compromised update being installed which can cause vulnerabilities in other parts of the security system of the application.

5 In depth analysis

5.1 Overview

In this section we take a look at 10 popular messaging apps and analyse the cryptographic primitives used to determine if they are vulnerable to quantum attacks. The apps were selected based on popularity by looking at downloads on the Google Play store. The apps that were selected also needed to have official documentation of the types of cryptographic primitives. This left out a few messaging apps such as WeChat and SnapChat where the companies hadn't officially documented what they had used for their security. In this analysis we will mostly focus on the threats based on quantum attacks and not look too much into classical attacks, although if there are any basic concerns they will be mentioned. We will look at all security used within the messaging apps which includes key exchange, message encryption, authentication and hashing.

5.2 WhatsApp

WhatsApp is a messaging app written in Erlang and owned by Facebook. It is is one of the most popular messaging apps with over 5 billion installs on the Google Play store. It provides users with end-to-end encryption for all chats, including group chats. With WhatApp being the most popular messaging app, it puts itself at high risk of an attack. The cryptographic primitives used in the end-to-end encryption system are as follows[16]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	Curve25519	HMAC-SHA256	AES-256	SHA-256

For key exchange WhatsApp uses ECDH using Curve25519 for its elliptic curve. This provides it with good enough security against classical attacks. ECDH however can be vulnerable to a quantum attack using Shor's algorithm. As previously mentioned the conciseness on an attack on the key exchange protocol will mean that the remaining security will not be effective as the communications will not be secure.

As for authentication, HMAC-SHA256 is used in this case. SHA-256 is the underlying hash function used in the HMAC and it is also used else in WhatsApp for hashing. SHA-256 is cryptographically secure against Grover's algorithm and therefore is not something that an attack will be able to break.

AES-256 is used for encrypting messages sent between two users. While AES-128 isn't secure against Grover's algorithm, AES-256 is and provides good enough security to protect from an attack.

End-to-end encryption is only enabled on supported devices and can be bypassed is using a device that doesn't support it. However, it can mean that some of the primitives mentioned may not be used in a chat that isn't end-to-end encrypted, which can have an impact on the security. The white paper written by WhatsApp[16] only provides an overview on end-to-end encrypted chats and an exploit using an outdated device with lower security could be a possibility. It is enabled by default on supported devices so the security mentioned will be in the majority of the chats.

5.3 Facebook Messenger

Facebook Messenger is a popular messaging app that has been developed from the Facebook social media site. It has had over a billion installs on the Google Play store and can be used through a web application and the Facebook social media site. It supports end-to-encryption, known as secret chat, however is optional and needs to be enabled. Facebook Messenger can be used by all users of the Facebook social media site, making it a very popular application. Due to this it has a greater chance of being attacked. The current security of the app isn't great as it only provides users with optional end-to-end encryption which needs to be enabled for each chat. The cryptographic primitives used in secret chat are as follows[17]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	Curve25519	HMAC-SHA256	AES-256	SHA-256

Facebook Messenger and WhatsApp are developed by the same company, so it isn't much of a surprise that they have the exact same underlying cryptographic primitives. As previously mentioned ECDH can be broken by Shor's algorithm on a quantum computer. This can then lead to the whole security system of the application being compromised.

Again SHA-256 is used for the HMAC and hashing throughout the application and is considered to be secure against a quantum attack.

The message encryption used is AES-256 which has long enough key lengths to protect from an attack using Grover's algorithm.

Even though Facebook messenger provides the same security as WhatsApp it doesn't provide end-to-end encryption by default. To enable it you need to enter a "secret chat" with the desired user to enable it. The primitives were taken from the white paper by Facebook[17] and only provides information on security of "secret chats". Therefore some of the encryption methods mentioned may not be in use. It is more likely for there to be a larger amount of non end-to-end encrypted chats as making it optional means that a large portion of users will not use it or may not even know about it.

5.4 iMessage

iMessage is developed by Apple and is only available for Apple devices. It is an app native to Apple devices and therefore is considered to be one of the most popular messaging apps. Due to iMessage's large user base it makes it a target for an attack. End-to-end encryption is enabled in the messaging app which uses the following cryptographic primitives[18]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
TLS	P-256	ECDSA 256/ HMAC-SHA256	RSA-1280/ AES-128/ ECIES	SHA-1/ SHA256

From this table we can see that TLS is used as the key exchange protocol. TLS (Transport Layer Security) is a protocol used for communication between two parties. DH is used to securely generate the unique session keys used for encryption and decryption in the TLS protocol. ECDH is often used in place of DH and in this case ECDH is used with the P-256 elliptic curve. As ECDH isn't secure against quantum attacks using Shor's algorithm, this means that TLS is vulnerable too. We have mentioned that an attack on the key exchange protocol used has a major impact on the rest of the security rendering it useless.

ECDSA is used with the P-256 elliptic curve as the method for signing messages sent by users to authenticate themselves. As we know ECDSA uses the discrete logarithm problem as a means for its security. Therefore an attacker with a sufficient quantum computer that can run Shor's algorithm efficiently will have no trouble breaking the security of the signature. They will then be able to sign their own messages claiming to be other people and destroy the integrity of the message. HMAC-SHA256 is used as a means of constructing a 40 bit value of the sender and receivers public key and plaintext, this is then combined with a randomly generated 88 bit value to create an authenticated key to use for the AES message encryption. HMAC is secure against attacks using Grover's algorithm and doesn't weaken the security of the app.

There are lots of different methods for message encryption used in iMessage. AES-128 is used to encrypt the main message body that is sent. RSA-1280 is used to encrypt the AES message key. In this case the AES message encryption doesn't need to be broken in order to break the security. Only the RSA encryption needs to be broken, which can be done when using Shor's algorithm. From there you will then have the key used in the AES encryption. Nevertheless, AES-128 is susceptible to an attack

using Grover's algorithm due to its shorter key size, so even if the encryption key remained secure, it would be possible to break the message encryption itself. ECIES (Elliptic Curve Integrated Encryption Scheme) is also used in older devices. The underlying security is dependent on elliptic curves and the discrete logarithm problem[19]. Both of these can be broken by broken by Shor's algorithm, thus making ECIES a vulnerable primitive to a quantum attack. An attacker can easily downgrade their device to ensure that ECIES is used over RSA and AES.

As for hashing, SHA-1 is used to form a hash of the encrypted message and its key. SHA-1 doesn't need to be attacked using a quantum computer as it already has vulnerabilities in classical security. It has been official deprecated by NIST since 2011[20]. This can lead to the an attacker modifying the message without the other two parties knowing. SHA-256 is used in the HMAC-SHA256 which is secure against quantum attacks.

At every point of security used in iMessage there is a vulnerable primitive used, whether it be to a classical or quantum attack. This is a big security issue as current technologies make the application vulnerable. This threat is amplified with iMessage being one of the top most used messaging apps. By breaking one layer in the security, the entire security of the application falls apart. The attack may be on the key exchange in which all proceeding traffic can be decrypted. It could also be on the RSA encryption used to encrypt the encryption key used for message encryption. It will not matter how secure the algorithms are for message encryption if an attacker is able to break the encryption to get the key.

5.5 Skype

Skype is a very popular messaging app developed by Microsoft. It is often used for its video conferencing features, which is often used in business. A dedicated Skype for business application has been developed for this reason. It has had over a billion downloads on the Google Play store on Android devices alone and commonly used on desktop devices. Skype was one of the first desktop applications that provides users with voice and video calling, making it very popular. Its large user base makes it a target for attacks. Skype uses the following cryptographic primitives[21]:

Key Exchange	Authentication	Message Encryption
TLS	RSA-1536/2048	AES-256

TLS is used for the key exchange in Skype. It is unclear as to whether ECDH or DH is used in the TLS protocol, however this doesn't make a difference to a quantum attack. This means that TLS is vulnerable to quantum attacks. An attack on the key exchange protocol can lead to the

remaining security to be broken, as the attacker can follow the communication between two parties to get all keys exchanged.

RSA-1536 and RSA-2048 is used to authenticate users between two parties. The security of RSA is broken by Shor's algorithm and can be run in polynomial time. Therefore the difference in security between RSA-1536 and RSA-2048 is negligible. An attack on this can lead to messages being impersonated by authenticating as someone else by breaking the authentication scheme.

AES-256 is used to encrypt messages within Skype. AES uses 256 bit key sizes and therefore is secure against an attack using Grover's algorithm.

An attack on the authentication scheme of Skype can have damaging effects on businesses using it for business related communications. The integrity of messages sent would be lost and there would be little trust in the identity of the sender.

5.6 Viber

Viber is a very popular messaging app with over 500 million installs on the Google Play store it provides users with end-to-end encrypted messaging and voice calling. The end-to-end encryption system uses similar concepts from the Signal messaging app protocol. End-to-end encryption is only enabled for devices that have the latest version supporting end-to-end-encryption. The following technologies are used within Viber[22]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	Curve25519	HMAC-SHA256	Salsa20-128	SHA-256/ MD5

Viber uses ECDH with Curve25519 for its key exchange protocol. As we know ECDH can be attacked using Shor's algorithm on a quantum computer. An attack on the key exchange protocol can lead to the remaining security becoming compromised.

HMAC algorithms are dependent of the security of the hashing algorithm used. In this case SHA-256 is used for the hash and for the authentication used in the HMAC algorithm. As SHA-256 is secure against attacks using Grover's algorithm, the authentication and hashing security isn't compromised.

The encryption scheme used for message encryption within Viber is Salsa20-128. Salsa20 is a stream cypher and can therefore be attacked using a brute force algorithm such as Grover's algorithm. Grover's algorithm reduces Salsa20-128's 128 bit security to 64bits of security, which is reasonably low enough level of security in order to break.

Therefore, to a quantum attack on Vibers message encryption an attacker will be able to compromise the security and decrypt the messages sent.

We have already mentioned SHA-256 being used as one of the hashing algorithms used in Viber. However, MD5 is also mentioned in the Viber white paper[22] which is used for signing files sent withing the messaging app to verify transmission integrity. MD5 is compromised hashing algorithm against classical security and has been deprecated. This opens users up to attacks where their files sent may be verified with an MD5 hash but may have been modified by an attacker. An attacker can exploit collisions in the MD5 hashing algorithm to make it seem as though the file is unchanged. This could then lead to a greater compromise if malware is placed in a file that is seemingly safe.

Viber is only end-to-end encrypted for devices using a version of the app that supports end-to-end encryption. An attacker is able to downgrade the version in order to reduce the security of the messaging app. The white paper only mentions the security used for their end-to-end encrypted supported app[22].

5.7 Telegram

Telegram is a cross platform messaging app with over 100 million installs on the Google Play store. Its client side code is open source, however the server side code is not. End-to-end encryption is supported but has to be enabled in each of the chats and can't be used within group chats. It uses the following technologies for its security[23, 24]:

Key Exchange	Authentication	Message Encryption	Hashing Algorithm
DH	RSA	AES-256	SHA-256

Telegram uses DH as its method of key exchange protocol. As mentioned before DH is vulnerable to a quantum attack using Shor's algorithm. An attack on the key exchange protocol severely weakens the remaining security. Knowing the initial key exchanged will mean that the remaining traffic can be decrypted.

RSA is used to authenticate messages sent between users and authenticate users connecting to Telegram servers. This primitive underlying security can be broken in polynomial time using Shor's algorithm and isn't quantum resistant. This opens the messaging app up to quantum attacks, meaning the authenticity of messages will no longer be valid.

AES-256 provides a sufficient key size to protect from quantum attacks using Grover's algorithm. SHA-256 is also able to survive a quantum attack.

The cryptographic primitives used within Telegram are only defined to be available when using end-to-end encryption. It is unclear what security is used for non end-to-end encrypted chats. The security defined by Telegram[23] is only for "secret chats" which enables end-to-end encryption. A "secret chat" is enabled on a per chat basis meaning that it needs to be manually enabled by the user for each chat. This means that a large portion of chats aren't end-to-end encrypted and may contain weaker security, which can open it to different vulnerabilities.

5.8 Signal

Signal is an open source messaging app and is one of the only messaging apps we found to be fully open source. The application and server for the messaging app are all open source allowing for more clarity on how it works. It is one of the most thoroughly documented messaging apps we analysed as the main focus of the app is on security. End-to-end encryption is enabled by default for all chats. On the Google Play store it has had over 10 million installs. It contains the following cryptographic primitives[25, 26, 27, 28]

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
X3DH	Curve25519	XEdDSA	AES-256	SHA-256

Signal uses X3DH[27] (Extended Triple Diffie-Hellman) which is an adaptation of the DH key agreement designed by Signal. It is designed to work asynchronously to allow communications where one of the parties is offline but has published information to a server. The other party is then able to use the information to send encrypted data. It based off the Diffie-Hellman protocol and has the same fundamental mathematics behind it, therefore it is vulnerable to quantum attacks using Shor's algorithm. An attack on the key exchange protocol can lead to the remaining security being compromised and therefore is more likely to be targeted in an attack.

XEdDSA[28] developed by Signal is used for authentication. It allows the use of a single key pair for signatures. It is based on EdDSA (Edwards-curve Digital Signature Algorithm) which uses elliptic curves and the discrete logarithm problem to provide strong security against classical security. However, discrete logarithms are computationally feasible for a quantum computer able to run Shor's algorithm. This means that XEdDSA is vulnerable to quantum attacks and the integrity of messages would be lost in the presence of a quantum computer able to run Shor's algorithm.

AES-256 and SHA-256 are used for the message encryption and hashing, respectively. Both primitives can be attacked using Grover's algorithm, but

provide enough security to not be broken.

Signal is very focused on its security and has developed some of its own standards to ensure that it is the most secure it can be. The Signal protocol is adopted by other messaging apps due to its good security. While it is susceptible to quantum attacks in the future it is likely that Signal will be one of the first apps to implement quantum safe technologies, in place of the key exchange protocol and authentication schemes when sufficient ones are defined.

5.9 Wickr

Wikr's main focus is on security and provides end-to-end encryption and allows for messages to be sent with an expiration time. Wickr offers communication packages for companies allowing for conference calling, screen sharing and file sharing/storage. It has over 5 million installs on the Google Play store. Wikr uses the following technologies for its security[29]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	P-521	ECDSA	AES-256	SHA-256

Wickr uses ECDH with the elliptic curve P-521 as its key exchange protocol. It provides strong security against classical attacks which can't do large integer factorisation in polynomial time. A quantum computer that is able to run Shor's algorithm however can perform large integer factorisation in polynomial time. Against a quantum attack ECDH is vulnerable and can weaken the remaining security on the messaging app and compromise conversations between users.

ECDSA is the authentication algorithm used for signing messages to protect the integrity of sent messages. While ECDSA is secure against classical attack a quantum attack using Shor's algorithm can break its security. An attack on ECDSA can mean that an attacker is able to sign messages with the signature used by other users and therefore impersonate others.

AES-256 provides a large enough key size to protect from a quantum attack using Grover's algorithm, so the message encryption is secure from attacks. SHA-256 is used for the hashing algorithm in Wickr. It use within Wickr is to ensure that a message payload hasn't been modified in transit by an unknown third party. Sha-256 is secure against a quantum attack using Grover's algorithm.

Wickr is another company who have a big focus on security for their users. An attack on Wickr can have a large effect on organisations who use it for conferencing and file sharing.

5.10 BBM Enterprise

BlackBerry Messenger (BBM) was a popular messenger app exclusive to BlackBerry devices. BlackBerry then later released BlackBerry Enterprise which is a cross platform messaging app. It has had over a million installs on the Google Play store and is a native app on BlackBerry devices. It allows for optional end-to-end encryption and uses the following technologies to do so[30]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	P-521	ECDSA	AES-256	SHA-256

BBM uses a P-521 elliptic curve in the ECDH protocol used for its key exchange. ECDH is vulnerable to quantum attacks using Shor's algorithm which can have great effects on the other implemented security of the messaging app, as an attacker will be able to follow the traffic from the initial key agreement onwards. From this they would then be able to decrypt the AES-256 message encryption even though AES-256 is a quantum secure algorithm.

An attacker with a good enough quantum computer would be able to break the authentication scheme used in BBM. ECDSA is used to sign the messages of a chat which allows users to confirm the integrity of the message. A loss in integrity can cause users to be unknowing as to who the sender of a message actually was.

Grover's algorithm is the best quantum attack against SHA-256 but doesn't reduce the security of it enough for it to be broken. The hashes used in BBM are therefore safe against a quantum attack.

BlackBerry Messenger's end-to-end encryption and the standards mentioned above are used for every chat making it as secure as it can be. A vulnerability in the security of the app can affect lots of users due to it being a native app on BlackBerry devices.

5.11 Wire

Wire is end-to-end encrypted messaging app developed by a team including the co-founder of Skype with many of the employees also being from Skype. Its client side code is open source and provides end-to-end encryption for all its features by default. It uses the following cryptographic primitives for its security[31]:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
ECDH	Curve25519	HMAC-SHA256	AES-256/128	SHA2-257

Wire uses ECDH with Curve25519 for its elliptic curve to provide good security for its key exchanges against classical attacks. Quantum attacks against ECDH are possible and will cause the remaining security used in the messaging app to be exposed.

HMAC-SHA256, SHA2-257 and AES-256 are used within the Wire messaging app and would need to be brute forced using Grover's algorithm to provide the best chance of breaking them using a quantum computer All these primitives are however not able to be broken and provide post quantum security for their uses in the messaging app.

While AES-256 is used for message encryption and is quantum safe, AES-128 is also used for message encryption. AES-128 is only used to encrypt data locally stored on a users device. While this makes it harder for an attacker to break as they will need to gain access to the devices local storage in order to get the encrypted messages it's not impossible. Once an attacker has the AES-128 encrypted messages, with a good enough quantum computer Grover's algorithm can be used to break the security and decrypt the messages.

6 Conclusion

Overall, we can see that quantum computers will have a major impact on currently implemented security systems. Messaging apps are one of the most frequently used applications by users all over the world. The impact of an attack on the underlying cryptosystems of a messaging app can be catastrophic and render the security useless. Algorithms such as Shor's and Grover's make use of quantum computers to run with better complexities than they could run with on classical computers. These algorithms can break the fundamentals mathematics behind some of the cryptographic primitives used within popular messaging apps.

As we have seen all of the ten messaging apps analysed had at least one vulnerability to quantum attack. All of the vulnerabilities were on the key exchange protocol. The key exchange protocol holds up the remaining security of the application and therefore with it being compromised, due to a quantum attack, the security of the messaging app is compromised. An attacker is able to follow the traffic between the parties involved in the exchange to obtain future keys and bypass the need to break the security behind any of the message encryption as they will be able to see the keys used.

A large portion of applications used ECDSA for signing messaged to proved integrity. This is broken by a quantum attack which can allow for an attacker to impersonate users on the messaging app. RSA was also used for authentication and in one case it was used for encryption of the message encryption keys in the iMessage application. Similarly to ECDSA it is also broken by Shor's algorithm and will have the same consequences. With the use of RSA in iMessage however it would have a greater impact allowing an attacker to decrypt the message encryption key and therefore decrypt messages sent.

All messaging apps with exception of Viber used AES for message encryption. Most of the AES key lengths were sufficient enough to protect from an attack using Grover's algorithm on a quantum computer. iMessage used 128 bits for its AES key size and Viber also used a 128 bit key for its Salsa20 encryption. Wire also used AES-128 for message encryption but only for its locally stored data. A 128 bit key isn't big enough to protect from an attack using Grover's algorithm and to secure the message encryption larger key sizes are needed.

As for hashing algorithms, none of the ones used are vulnerable to only quantum attacks. Instead some used are vulnerable to classical attacks. MD5 used in Viber and SHA-1 used in iMessage are both deprecated hashing algorithms and can be exploited causing collisions. Integrity of files on these platforms are lost where these hashes are used.

It is difficult to rank the security of messaging apps with all of them being compromised at the key exchange level. For most of the platforms the end-to-end encryption is optional. This means that the security used is potentially worse than that documented which can lead greater vulnerabilities in the messaging applications. Having end-to-end encryption enabled by default means that more users are more secure and only a small portion of the apps analysed do. Other methods to remove end-to-end encryption such as downgrading the device or application version can be used to bypass some of the security features used in end-to-end encryption. The following is an overview of the security of messaging apps against quantum and classical attacks:

Messaging App	Risk
WhatsApp	High
Facebook Messenger	High
iMessage	Very High
Skype	High
Viber	Very High
Telegram	Very High
Signal	High
Wickr	High
BBM Enterprise	High
Wire	High

An ideal messaging app with using current cryptographic primitives will use the following:

Key Exchange	Elliptic Curve	Authentication	Message Encryption	Hashing Algorithm
X3DH	Curve25519	HMAC-SHA256	AES-256	SHA-256

It should enable end-to-end encryption by default and not allow for older versions to be used without it. It uses similar cryptogrphic primitives to WhatsApp and Facebook and uses X3DH, used in Signal, for its key exchange.

This would have a risk level "medium" as it still uses X3DH which is vulnerable to a quantum attack. A quantum secure messaging app should include a post quantum key exchange protocol, however there aren't any that are suitable to be used inside a messaging app.

References

[1] WhatsApp. "Two Billion Users – Connecting the World Privately". In: *WhatsApp Blog* (February 12, 2020). URL: https://blog.whatsapp.com/10000666/Two-Billion-Users--Connecting-the-World-Privately.

[2] WhatsApp. "Protecting our users from a video calling cyber attack". In: *WhatsApp Blog* (). URL: https://faq.whatsapp.com/help/video-calling-cyber-attack.

[3] Vasileios Mavroeidis et al. "The Impact of Quantum Computing on Present Cryptography". In: *International Journal of Advanced Computer Science and Applications* (). URL: https://arxiv.org/pdf/1804.00200.pdf.

[4] Kerry et al. *Digital Signature Standard (DSS)*. Tech. rep. National Institute of Standards and Technology, July, 2013, pp. 22–25. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[5] Fabrice Boudot et al. *795-bit factoring and discrete logarithms*. Tech. rep. cado-nfs, December 2, 2019. URL: https://lists.gforge.inria.fr/pipermail/cado-nfs-discuss/2019-December/001139.html.

[6] Whitfield Diffie and Martin Hellman. "New directions in cryptography". In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654. URL: https://ee.stanford.edu/~hellman/publications/24.pdf.

[7] Kerry et al. *Digital Signature Standard (DSS)*. Tech. rep. National Institute of Standards and Technology, July, 2013, pp. 15–21. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[8] National Institute of Standards and Technology. *Advanced Encryption Standard (AES)*. Tech. rep. National Institute of Standards and Technology, November 26, 2001. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

[9] Robert Chesebrough Gael Hofemeier. *Introduction to Intel® AES-NI and Intel® Secure Key instructions*. Tech. rep. Intel, July 26, 2012. URL: https://software.intel.com/sites/default/files/m/d/4/1/d/8/Introduction_to_Intel_Secure_Key_Instructions.pdf.

[10] Niels Ferguson et al. "Improved cryptanalysis of Rijndael". In: *International Workshop on Fast Software Encryption*. Springer. 2000, pp. 213–230. URL: https://www.schneier.com/academic/paperfiles/paper-rijndael.pdf.

[11] Kerry et al. *Digital Signature Standard (DSS)*. Tech. rep. National Institute of Standards and Technology, July, 2013, p. 104. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.

[12] Daniel J Bernstein. "Curve25519: new Diffie-Hellman speed records". In: *International Workshop on Public Key Cryptography*. Springer. 2006, pp. 207–228. URL: https://cr.yp.to/ecdh/curve25519-20060209.pdf.

[13] H.Krawczyk, M.Bellare, and R.Canetti. *HMAC: Keyed-Hashing for Message Authentication*. Tech. rep. Network Working Group, 1997. URL: https://tools.ietf.org/pdf/rfc2104.pdf.

[14] Peter W Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". In: *SIAM review* 41.2 (1999), pp. 303–332. URL: https://arxiv.org/pdf/quant-ph/9508027.pdf.

[15] Lov K. Grover. *A fast quantum mechanical algorithm for database search*. 1996. arXiv: quant-ph/9605043 [quant-ph]. URL: https://arxiv.org/pdf/quant-ph/9605043.pdf.

[16] WhatsApp. *WhatsApp Encryption Overview*. Tech. rep. WhatsApp, December 19, 2017. URL: https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf.

[17] Facebook. *Messenger Secret Conversations*. Tech. rep. Facebook, May 18, 2017. URL: https://about.fb.com/wp-content/uploads/2016/07/messenger-secret-conversations-technical-whitepaper.pdf.

[18] Apple. *Apple Platform Security*. Tech. rep. Apple, Fall 2019, pp. 99–101. URL: https://manuals.info.apple.com/MANUALS/1000/MA1902/en_US/apple-platform-security-guide.pdf#page=99.

[19] Victor Shoup. "A proposal for an ISO standard for public key encryption (version 2.1)". In: *IACR e-Print Archive* 112 (2001). URL: https://eprint.iacr.org/2001/112.pdf.

[20] Elaine Barker and Allen Roginsky. "Transitions: Recommendation for transitioning the use of cryptographic algorithms and key lengths". In: *NIST Special Publication* 800 (2011), 131A. URL: http://www.gocs.eu/pages/fachberichte/archiv/075-sp800-131A.pdf.

[21] Skype. *Does Skype use encryption?* Tech. rep. Microsoft. URL: https://support.skype.com/en/faq/fa31/does-skype-use-encryption.

[22] Viber. *Viber Encryption Overview*. Tech. rep. Viber. URL: https://www.viber.com/app/uploads/viber-encryption-overview.pdf.

[23] Telegram. *End-to-End Encryption and Secret Chats*. Tech. rep. Telegram. URL: https://core.telegram.org/api/end-to-end.

[24] Telegram. *How is the server authenticated during DH key exchange*. URL: https://core.telegram.org/techfaq#authentication.

[25] Trevor Perrin (editor) and Moxie Marlinspike. *The Double Ratchet Algorithm*. Tech. rep. Signal, November 20, 2016. URL: https://signal.org/docs/specifications/doubleratchet/.

[26] Moxie Marlinspike and Trevor Perrin. *The Sesame Algorithm: Session Management for Asynchronous Message Encryption*. Tech. rep. Signal, April 14, 2017. URL: https://signal.org/docs/specifications/sesame/.

[27] Moxie Marlinspike and Trevor Perrin (editor). *The X3DH Key Agreement Protocol*. Tech. rep. Signal, November 04, 2016. URL: https://signal.org/docs/specifications/x3dh/x3dh.pdf.

[28] Trevor Perrin (editor). *The XEdDSA and VXEdDSA Signature Schemes*. Tech. rep. Signal, October 20, 2016. URL: https://signal.org/docs/specifications/xeddsa/xeddsa.pdf.

[29] Chris Howell, Tom Leavy, and Joël Alwen. *Wickr Messaging Protocol*. Tech. rep. Wickr, 2017. URL: https://1c9n2u3hx1x732fbvk1ype2x-wpengine.netdna-ssl.com/wp-content/uploads/2019/12/WhitePaper_WickrMessagingProtocol.pdf.

[30] Blackberry. *BBM Enterprise*. Tech. rep. Blackberry. URL: https://docs.blackberry.com/content/dam/docs-blackberry-com/release-pdfs/en/bbm-enterprise/1-10/BBM_Enterprise_Security_Note.pdf.

[31] Wire Swiss GmbH. *Wire Security Whitepaper*. Tech. rep. URL: https://wire-docs.wire.com/download/Wire+Security+Whitepaper.pdf.