A study in Cryptography

By Kyriakos Sourmelis

> “One must acknowledge with cryptography no amount of violence will ever solve a math problem.”
> — Jacob Appelbaum, “Cypherpunks: Freedom and the Future of the Internet”

Cryptography is the study and practice of techniques used to secure communications between parties and avoid being looked upon by third party. Generally speaking, cryptography constructs and analyzes protocols that prevent any third party of having access to private data that might concern individuals or government bodies, and it is applied to information security to ensure data confidentiality, data integrity, authentication and non-repudiation. Modern cryptography relies heavily on advanced mathematics, computer science, physics, electrical engineering, and communications science.

The history of cryptography can be traced all the way back to 700 BC in Ancient Greece, in Sparta, were they used an encryption code called Scytale. The Spartan military used Scytale to send messages during battle to organize their troops. Both sender and recipient had a wooden rod of the exact same diameter and length. Each side sent their messages by tightly wrapping a piece of leather around the stick and wrote the message on it. The unwound leather was then sent to the recipient. The message could only be read when it was tightly wrapped around the Scytale of the recipient party. In case it fell to unwanted hands, they couldn’t read anything except disarranged letters with no meaning. This was the very first method of encrypting a message in history.

Ciphering wasn’t only used in Ancient Greece though. In Ancient Rome, Julius Caesar thought of a way to communicate with those close to him in a cipher that was since named after him, the Caesar Cipher. Caesar had the idea of shifting the alphabet either left or right and writing his messages in code. The letters would shift according to a number chosen by him and the whole correspondence will take place in a jumbled up alphabet. If, for example, the shift number chosen by the Caesar was 5 to the right, then this meant that in case he wanted to write the letter A, he would instead write the letter E. So a simple word like “Hello” would look like “MJQQT”. This made no sense to anyone who might sneak a peek into his private correspondence so the emperor’s letters were protected. The Caesar Cipher is a famous cipher and it is still used today, usually as a part of more complex schemes, such as the Vigenère cipher.

Fast forward to 1467 where this was the time when the first polyalphabetic substitution cipher was invented by a man called Leon Battista Alberti. Alberti was a humanist, author, poet, linguist and a cryptographer. This important discovery changed the course of encryption forever and set it to the path we all now know and use today. The Alberti cipher was comprised of 2 metal discs on the same axle, one inside the other, both of which involved a jumbled alphabet and a variable number in rotations given by the user. This technique introduced the polyalphabetic cipher as well as machine assistance encryption using a cipher disk. This was a huge step towards modern cryptography since the previous contribution to the field was the Caesar’s Cipher back in Ancient Rome. In fact, Alberti’s contribution is so important, that the renowned cryptography historian David Kahn called him the “The father of Western Cryptography” and credits him with advances like the earliest Western exposition of cryptanalysis, the invention of polyalphabetic substitution, and the invention of enciphered code.

We had to wait for more than 300 years to take another step forward in cryptography. This time was with the aid of inventor Thomas Jefferson, in 1795. He invented a system that was made up of 26 cylindrically wooden pieces threaded onto an iron spindle. The letters of the alphabet were inscribed on the edge of each wheel in random order. The Jefferson disk or Jefferson wheel as it is also known, was working by arranging the letters to display a message and then the sender would copy one of the other rows and send it to the recipient. The text was unreadable by any third party not in possession of the invention but to the recipient it meant to rearrange the letters as shown in the text. Thus, one of the 26 rows contained one row that it was readable and the message was then relayed. Despite being an easy code to crack when used for more than one line of text, this technique was used by the US army from 1923 to 1942 and it was referred to as the M14. Commandant Étienne Bazeries independently invented the same system about a century later.

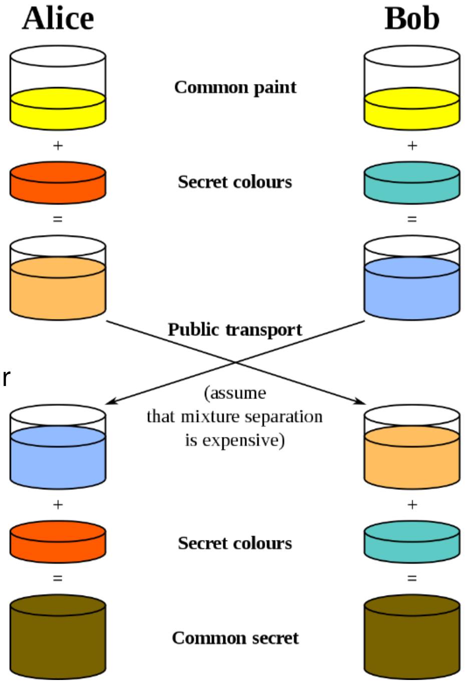
One of the biggest events in our history was the Second World War, which brought catastrophe, deaths and took a toll to the world wide economies and to the people. During those troubled times, the Axis powers and especially the Nazis, we're using a machine called Enigma to try and encrypt the messages they sent to their troops or to other governments in order to secure them against the prying eyes of the Allied forces. Enigma consists of a series of related electro-mechanical rotors that scramble the Latin alphabet. This encrypts the plain English text using lights that go on if they are used. This, in combination with a list of encryption keys that was distributed in advanced to the Axis forces, resulted in a message that made no sense to anyone looking at it. Enigma was a very intricate machine and it really puzzled scientists that were trying to break the code in order to have an edge over the enemy. In the UK, a team consisting of scientists was trying actively to break the code. That team had a member that went on to be known as the father of modern computer science. The great Alan Turing was a member of the Bletchley Park team that was trying to decrypt the Enigma machine. Through his efforts, his highly advanced intellect and the charisma in discovering and viewing stuff other people either couldn't or ignored, the team managed to break the code and identify Wehrmacht's every move giving the Allied forces the edge and advantage that they needed to defeat the enemy.

Before reaching the present and looking into algorithms that are in use today, we take a look at the generation of the first computer password. Back in 1961, the Massachusetts Institute of Technology (MIT) created the first computer password for the use of their Compatible Time Sharing System (CTSS). This was the first time we are able to witness a username and password being used, and the credentials going through the process of user authentication. The CTSS was also the first system to experience a password breach, albeit accidental. In 1966, a software bug jumbled up the systems welcome message and its master password file, so that anyone who logged in was presented with the entire list of the user's passwords.

Our last stop in the history of encryption is the discovery of the first ever encryption algorithm, the one that started the field of cryptography and generated the need for more powerful processes to use for securing sensitive data. In 1979, the National Agency of Standards created the DES algorithm, which stands for Data Encryption Standard. It uses a 56-bit key for encryption and at the time it was so powerful that even supercomputers of that era couldn't crack it. The DES encryption was the standard for almost 20 years and it found many applications through that time. In fact it was so powerful that the National Security Agency of America, the NSA, used it to encrypt their data and communications. As time went by and computers became more and more powerful, the DES algorithm was deemed too weak. It was easily breakable and the retrieval of information was very easy; which brings us to today…

Modern cryptography is based on the concept of the utilization of encryption keys. There is no ciphering being used as it is easily reversible leaving sensitive information exposed. There are two categories for cryptography: symmetric and asymmetric. The categories are characterized by the length of the keys that are used to encrypt the data and the communication between the parties.

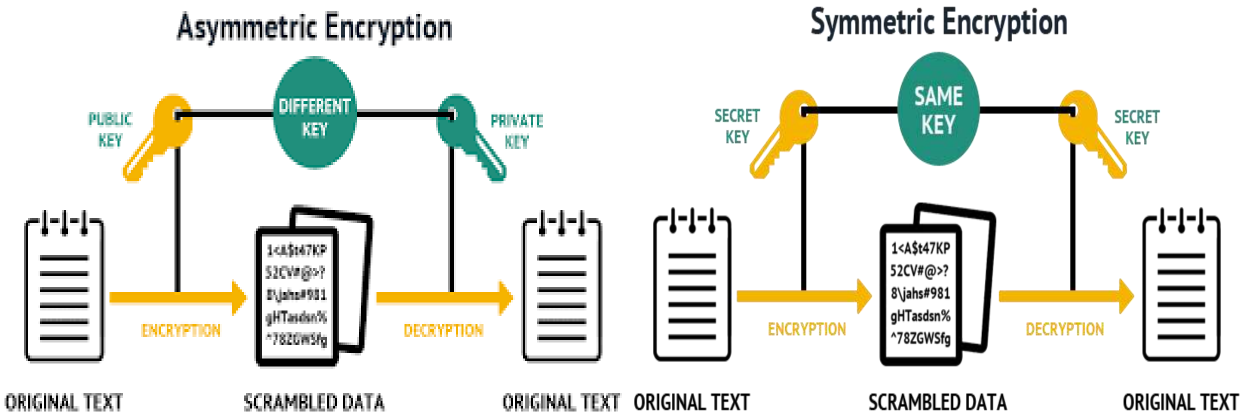
Symmetric key cryptography uses a shared key to encrypt or decrypt the text. This key can be the same or a slight variation of it during encryption and decryption to distinguish the two between them. Let's assume that Bob has a box that keeps his secrets secured with a lock. The lock uses a key that can lock and unlock the box, so if Bob wants to keep his secrets and conversations private, he puts it in the box and locks it so only him or someone else with a copy of his key can open the box. Since both parties that the conversation takes place in between have possession of the same key, this algorithm is not as safe as we want it to be. Despite this drawback, this type of encryption is used by 3DES (a very powerful algorithm and successor to the DES encryption algorithm), Blowfish and Serpent. The reason behind its usage is because it is easier to encrypt bulks of data using one key and to decipher them using the same one. The problem lies with the two parties sharing the encryption key. This was solved using the Diffie – Hellman key exchange protocol, which utilizes a non-secure channel to exchange a secure key, ensuring that the exchange from that point onwards is encrypted. DH key exchange is used today to transmit a shared key for encryption algorithms and it is also used during the establishment of the TLS protocol used in communications over the Internet. The following example shows how the key exchange takes place over the non-secure, public channel (illustrated here): Bob and Alice want to exchange a secret colour. They both choose a colour in common. In this case, the colour is yellow. This information is public. Then each one of them chooses a secret colour (Alice chooses red and Bob chooses green). These colours are not public and should remain with Bob and Alice. Each one of them mixes the common, public colour with their respective secret colour. Then, they exchange the mixed colour and each one mixes the received colour with their respective secret colour. The result in both sides is the same yellowish brown colour, which is the symmetric key in this example.



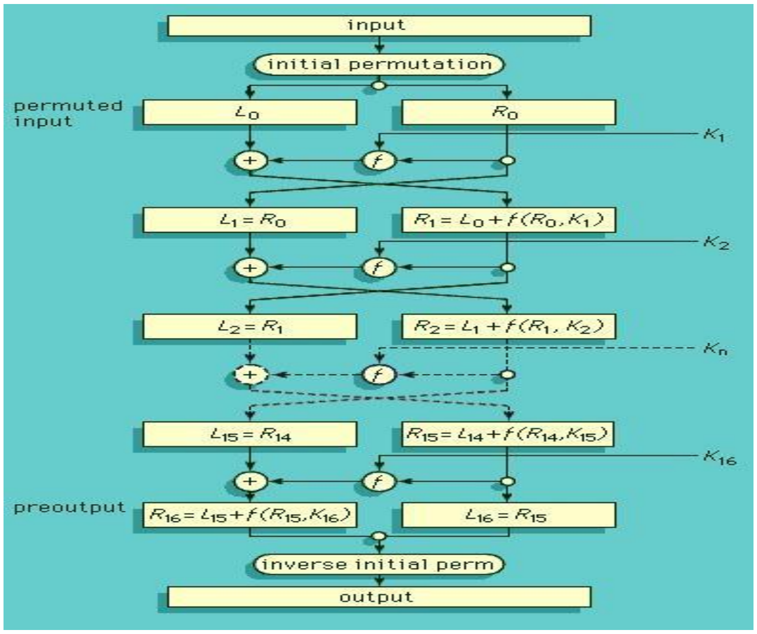
The asymmetric key encryption is a relatively new method of encryption. It uses a pair of two (2) cryptographic keys to encrypt the data. The first key used is called a “public key” and the second one is called “private key”. In order to demonstrate how this algorithm works is better if we look at an example. In our example, Bob and Alice want to have a private conversation. They both have the same public key and different private keys. Bob wants to send a message to Alice but he wants to keep it away from prying eyes. In this case, the clear message is encrypted with the public key, sent over to Alice and it is decrypted with her private key. Despite its simplistic nature, this algorithm of encryption and decryption is very robust. Each key is produced through mathematical problems that produce one-way functions. This way the reverse engineering of the private keys is virtually impossible, thus ensuring the security of the data. The keys produced from this one way function are such that the combination of Bob’s private key combined with the public key they both share, and Alice’s private key combined with their public key, produce a result that is known as their “shared key”. Asymmetric keys are used widely nowadays with TLS, DNS and digital certificates being the prominent users of this technology.

While both symmetric and asymmetric key algorithms have their own merit, they are not without disadvantages. Symmetric key encryption is much faster in implementation and it does not require a lot of resources when compared to asymmetric key encryption, and it is relatively simple in understanding and materialization. The only thing that has to be agreed upon between users is the use of their public key.

However, the exchange of keys using the DH key exchange protocol mentioned before is not without its problems. Should someone eavesdrop on the channel (remember it is a non-secure channel) then the secret is out. On the other hand, the asymmetric key encryption has the additional advantage of providing verification with the use of digital signatures, and while their public keys are indeed public, their respective private keys are held private. However, asymmetric key encryption requires much more resources to be produced; it is slower in verification, decryption and encryption and in case one of the parties loses their private key, then the received message cannot be decrypted.



There are a variety of encryption methods in use today. The most common one, the one that started it all, is the DES algorithm. The DES algorithm works by getting the clear text and then produces a block cipher by encrypting it. This produced result is subject to further encryption and combining two or more transposition ciphers or substitution ciphers, thus allowing for a more secure encryption as a result. The process takes place over 16 repetitions of substitution and transposition processes. Initially, the block size consists of 64-bits and the key which controls the transformation is of the same size. The user on the other hand can only choose 56 of these bits which constitute his key. The remaining 8 bits are used as parity check byte and later totally terminated and excluded by the process. The picture on the side indicates how the events of the various permutations and transformations take place during one round of encryption. We can see in the picture that we have 2 sets of keys, L0 and R0, which consist of 32 bits each. With each iteration the keys change and are represented with R1 and L1, etc. to mark the iteration round. At the next step, R0 is going to take the value of L1 and R0 is going through a function and then it is added with L0; this is the value of R1, K is a sub-key with a length of 32 bits, and it is the same for all iterations added to the function that manipulates R0. In the next step, the same thing happens again, R1 is going to be the value of L2 and R1 is going through a function and then it is added with L1, thus giving us the value of R2. This process takes place 16 times or rounds, at the end of which the right and left keys are going to switch places. This means that L15 is going to become R16 and R16’s value will be L15 plus the outcome of the function, the right part is going to be L16. Finally, the output is going to be the result of a process called the inverse initial



permutation of those two keys R16 and L16, thus giving us a set (pair) of keys, with the same length of 64bits. The algorithm is so powerful that to brute force it you need to search 2 in the power of 56 different values (keys); it is like searching for a needle in a haystack of 72 quadrillion straws. However impossible that might sound, back in 1999 a DES search engine combined with 100,000 computers on the Internet found the key in 22 hours. A fascinating feat, don't you think? This signaled the end of an era for the most powerful known encryption algorithm at the time.

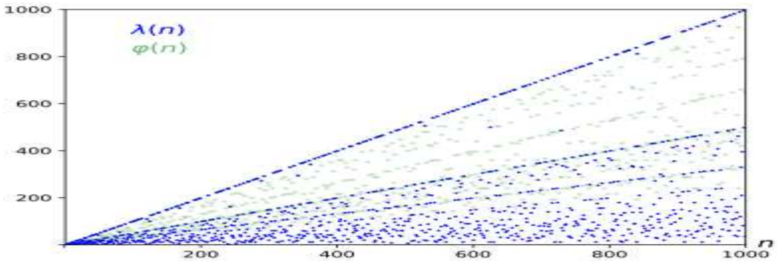
In 1977, Rivest, Shamir and Adelman, made another huge step for cryptography. They discovered an encryption algorithm that they named after their initials, the RSA algorithm. This algorithm is based on public key encryption and its main purpose is the factorization of very large prime numbers. The concept behind this algorithm is not very simple, but we will try to explain it using smaller numbers. Bear in mind that the basis of the RSA algorithm is that while the calculation done in the forward order are fairly simple, reversing them is almost impossible, and this is where the algorithm bases its strength. The algorithm's nature and birth comes from deep knowledge of mathematics and for that we need first to understand a few basic things used to produce the result of RSA. These mathematical concepts include the trapdoor functions, how prime numbers are generated, Carmichael's totient function and the processes that compute the public and private keys.

For our examples we will use small numbers to get the idea behind the mathematics involved as real numbers used in this algorithm are very, very difficult to calculate. So, firstly, let's start with the concept of the trapdoor function which relies on the idea of finding large numbers that are the product of multiplication of two large prime numbers. As an example let's take number 464,869. This number is the product of two prime numbers. Can you figure out which those two prime numbers are? Seeing in the other way around though is much easier. 619 * 751 is 464,869. Our prime numbers involved in this calculation are 619 and 751. To compute them in reverse however is not an easy feat. This calculation is what the mathematicians call a trapdoor function. To put things in perspective here, when we are doing this for the real RSA encryption, prime numbers can be up to 2048 digits long and they will produce keys that are 617 digits long. So, as a first step in encrypting a message, is to generate the keys. In order to do this, we will need 2 prime numbers (a) and (b). These numbers are selected with the primality test (an algorithm that can find large prime numbers). For our example we will use the relatively small numbers we used before. So, in our case, (a) is 619 and (b) is 751. The next step is to find the modulus (n) using this simple formula, n = a * b. If we did our calculations correctly, the resulting (n) should be 464,869. Now that we have our (n), we will use Carmichael's totient function, $\lambda(n) = \mathrm{lcm}\ (a - 1, b - 1)$. Despite this looking scary and complicated to process, please allow me to analyze it to further give you an insight into all of this. First $\lambda(n)$ is called Carmichael's totient for (n) and lcm is the lowest number that both (a) and (b) can divide into (this is known as the least common multiple). Let's put our numbers from the example above in the equation. We will get:

$$\lambda\ (464{,}869) = \mathrm{lcm}\ (619 - 1, 751 - 1)$$

Next step is to calculate the lcm. If done correctly, it should yield the value of:

$$\lambda\ (464{,}869) = 77{,}250$$



Now that we have Carmichael's totient, we can calculate the public key. Public keys are made of a prime number (e) in combination with n. Public key (e) can have any value between one and λ(n), which in our case is 77,250. The value of the public key (e) is generally set to 65,537. This is due to the fact that when larger numbers are chosen randomly, it makes encryption less efficient. To keep it simple let's use a small number as our public key, like 15. The output will be a ciphertext (c), which we derive it from our plaintext message (m) by using the public key (e) and the following formula:

$$c = m^e \bmod n$$

Let's replace the numbers in the formula and see how it works. The message that we want to encrypt is m, has the value of 5. This will give us the following equation:

$$c = 5^{15} \bmod 464{,}869$$

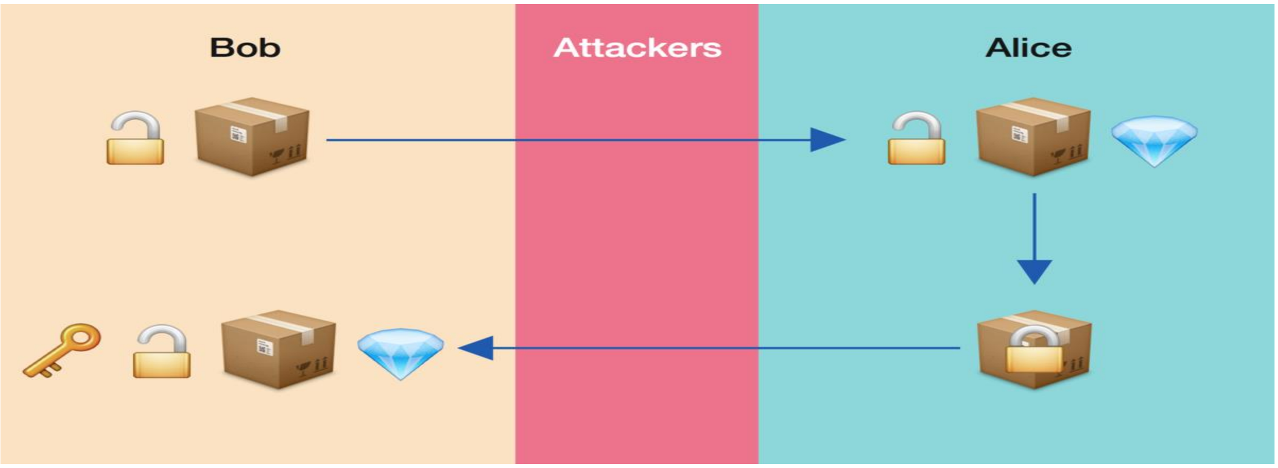
The result (c) is equal to 322,882. This is the encrypted or ciphered message that will be transmitted from one participant to the other. Now, each participant has a private key as we already said. The private keys are generated using the Carmichael's totient (n) and the value (d) that is derived from the following equation:

$$d = 1/e \bmod \lambda(n)$$

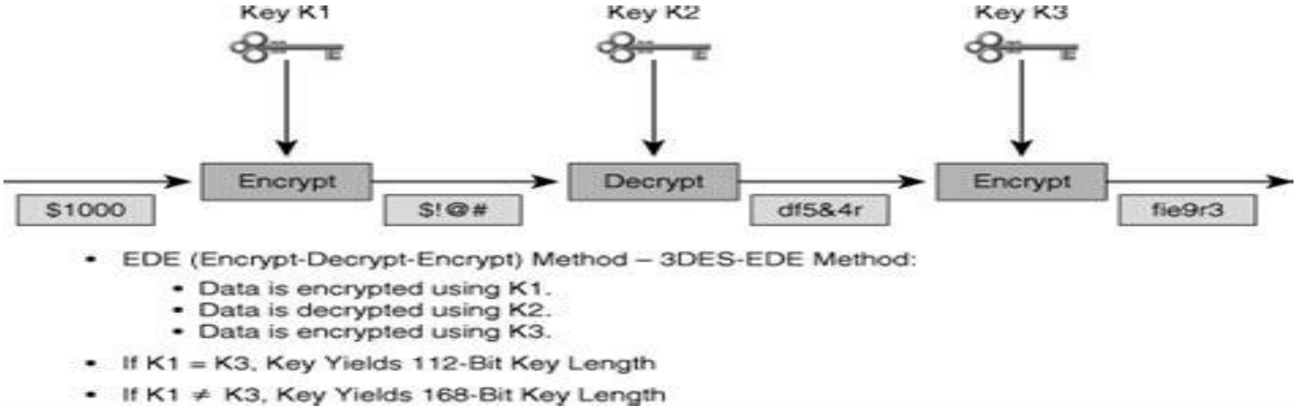
where 1/e is not 1/15 (which is our public key) but rather the inverse function of the public key. Let's put in our numbers from before in order to compute (d). So (e) is 15 and (λ(n)) is 322,882. In this case, (d) is equal to 43,051. With the use of d and n we can calculate two private keys, one for each participant and these are used to encrypt and decrypt the messages going back and forth. In order to decrypt a message encrypted with the private keys, we can use the formula:

$$m = c^d \bmod n$$

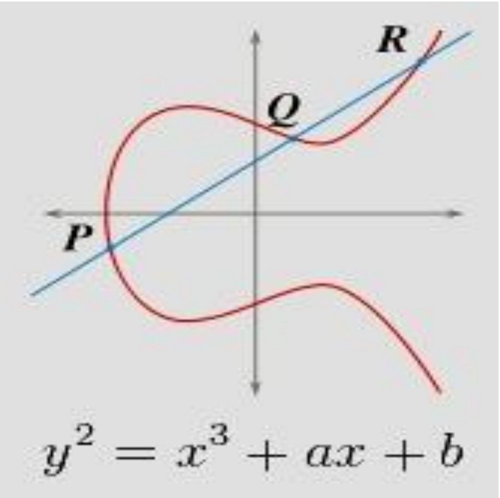
The way to calculate the outcome for the rest of the calculations henceforth is born from calculations that involve the movement of the point on a curve and how it reflects on the tangent at the specific point. This is what will later become the basis for elliptic curve cryptography, and it demands higher mathematics that is beyond the scope of this study. The previous example will give you an idea of how mathematics works in RSA encryption. It is worth noting that when the sender and the recipient have the same public key (in the case of RSA, they do) they can use it to encrypt the data, and when the data has been encrypted, it can only be decrypted by the participant's private key from the same key pair. Remember the public key cannot be used to decrypt the data as this is due to the properties of the trapdoor functions. This is a basic example on how RSA encryption works. Keep in mind the advanced and complex mathematics involved though and soon you can realize how difficult it is to crack and discover the keys involved in the exchange of information using the RSA algorithm.



With cryptography well under way and everyone looking for a way to make their data safer than everybody else's, a new idea came up borrowing elements from a fundamental principal. Welcome, 3DES encryption algorithm. Triple data encryption algorithm is like an upgraded version of the data encryption standard (DES). It uses the same principal as the original encryption standard but the loop goes over three times overall. This means that in total, 48 iterations are executed using three different keys, one per loop. The idea behind this three loops and different keys is to produce something that is literally impossible to decipher without the proper keys. The way it works is the following: Using key one (K1) we go through the DES algorithm encrypting the keys and trying to produce a pair. The second time we go through we use a second key (K2), different from the first one and instead of encrypting, we are trying to decrypt. Since the encryption happened with K1, the result when decrypting with a different key is something that it is not readable or understandable in any case or form. To finish the whole loop, we use a third key (K3), and we encrypt the data once more. There is a possibility that K1 and K3 might be the same, but this does not affect the efficiency of the algorithm. The only drawback from using the same keys for K1 and K3 is that the final pair of keys is of length 112 characters (bits). If, however K1 and K3 are different, then the resulting pair has keys with length of 168 characters (bits). The results yielded from this algorithm are so strong that are used nowadays from banks and armies from around the world. The image below shows how the algorithm operates as well as giving a brief example using the value of $1000.



- EDE (Encrypt-Decrypt-Encrypt) Method – 3DES-EDE Method:
  - Data is encrypted using K1.
  - Data is decrypted using K2.
  - Data is encrypted using K3.
- If K1 = K3, Key Yields 112-Bit Key Length
- If K1 ≠ K3, Key Yields 168-Bit Key Length



One of the most sophisticated and advanced techniques used in cryptography today, is the Elliptic Curve Cryptography. This method requires knowledge about complex mathematical meanings and equations, thus we will try to keep it as simple as possible. Elliptic curve cryptography was discovered by two researchers in 1985 and it was introduced and used widely in 2004 and 2005. The two scientists responsible for this discovery were Victor Miller and Neil Koblitz and it started as a different mechanism for applying symmetric key cryptography. The idea stems from the fact that elliptic curve cryptography is based on distinct logarithms, and these are much more difficult to test at equivalent key lengths.

But what are elliptic curves? Elliptic curves are a class of curves that meet certain mathematical criteria. A curve is considered elliptic if it is a smooth, planar curve and can take the Weierstrass form of:

$$y^2 = x^3 + ax + b, \text{ where } 4a^3 + 27b^2 \neq 0$$

An elliptic curve is shown in the picture here along with a line that satisfies the points P, Q and R which we will discuss next.

Elliptic curves don't look like geometric ellipses despite their name. They are the planar projections (2D projections) of 3D shapes that are created and manipulated with various concepts and they actually derive from the algebraic clash of equations described with:

$$ay^2 + by = cx^3 + dx^2 + ex + f \text{ where } \{a, b, c, d, e, f\} \in \mathrm{IR}$$

The points P, Q and R on the line that crosses the curve are points that satisfy several conditions. These points are rational solutions to the curve (meaning they have a value that even if it is decimal, it ends and does not continue to infinity) although point R is set to almost infinity, and that is why it is not used to satisfy the equation. The purpose of point R is to be the control of the group of solutions and it isn’t always obvious as to where it will intercept our curve. These solutions form what is known as a geometrically defined Abelian group of solutions. As a side note, a geometrically defined Abelian group is a group of numbers in topology (a branch of mathematics that deals with the properties of geometric objects that are preserved under continuous deformations) that will not affect the outcome of the equation when either of them is applied in whichever order to satisfy it. With the help of point R and the Mordell – Weil theorem, we can prove that this group is finitely generated, thus there are only certain solutions that belong to this group. In the picture above, we can see that point P and Q can also have mirror points (P’ and Q’) although not displayed in the image. These points are geometric opposites (they can be found by drawing a dotted line parallel to the y – axis from points P and Q to meet the curve again) giving two possible values for y for each x value that we input in the equation that describes the curve.

You may also have noticed that the line we chose intersects the curve in three points. What happens though if the line only intercepts the curve in two points (with one being at a tangent point) or one point (with the line being parallel to the y – axis)? In the case of only two points (one being the tangent) we assume that points P and Q match and are identical (same goes for P’ and Q’). In the case of only one point being present (line is parallel to the y-axis), then all three points (P, Q and R) match (as well as their respective mirror points) and that point is also the control point of the Abelian group.

The linking between the values of P and Q are used to produce the keys used by the elliptic curve cryptography algorithm. However, since the production of these keys requires Diophantine Equations and their best and fastest solution is an exponential one and derives from a discrete logarithmic problem, the discovery of these keys requires quantum computers and high knowledge of advanced and complex mathematics, making it one of the safest encryption algorithms known to us today.

While it is perfectly fine to use and investigate all the possible values in theoretical mathematics, in order to use them in cryptography we must first take into consideration that these numbers are going to be used on microprocessors to encrypt and decrypt messages. So firstly we have to limit the range of said numbers because it isn't logical or practical to have numbers almost near Infinity on 16, 32 or 64-bit microcontrollers. The elliptic curve is extended through to infinity and the vertical and horizontal axes are topped with a very large prime number (p) and in order to keep the results in range, we use the modulus operator (mod). For better understanding on how this algorithm works let’s take as an example a small number, like 250. The equation we will use is:

$$y^2 = x^3 - 4x + 5$$

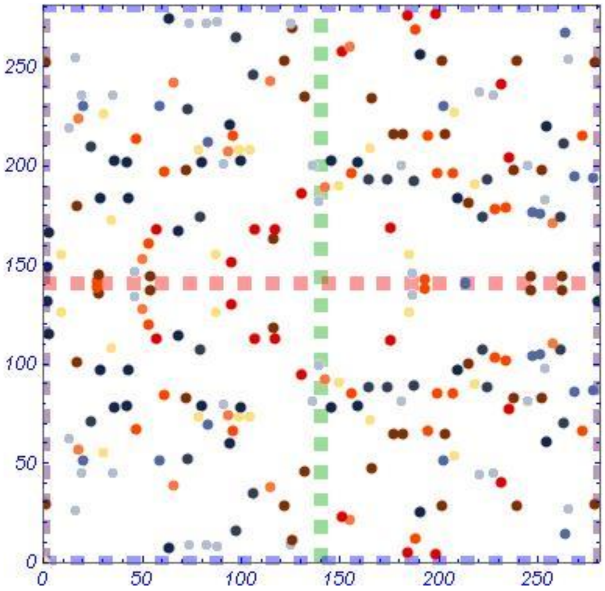
Using the modulus, the equation will be:

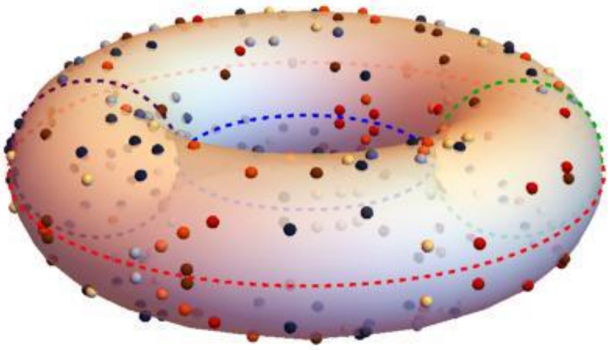
$$(y^2 - x^3 + 4x - 4) \bmod 250 = 0$$

with x and y being integers with values between 0 and 250. We will run all the possible combinations in the range we have defined (0 - 250) and when the remainder of the equation is 0 the point is added to a list of possible solutions. By using a computer program, we can see the following list that comprises of all the possible solutions to our equation, in the given range, in the following image:

● (1, 29)	● (1, 252)	● (2, 132)	● (2, 149)	● (3, 115)	● (3, 166)	● (9, 126)	● (9, 155)
● (13, 62)	● (13, 219)	● (16, 26)	● (16, 255)	● (17, 101)	● (17, 180)	● (18, 57)	● (18, 224)
● (19, 45)	● (19, 236)	● (20, 51)	● (20, 230)	● (24, 71)	● (24, 210)	● (27, 139)	● (27, 142)
● (28, 136)	● (28, 145)	● (29, 97)	● (29, 184)	● (30, 55)	● (30, 226)	● (34, 108)	● (34, 173)
● (35, 45)	● (35, 236)	● (36, 78)	● (36, 203)	● (42, 79)	● (42, 202)	● (43, 97)	● (43, 184)
● (46, 134)	● (46, 147)	● (47, 67)	● (47, 214)	● (50, 128)	● (50, 153)	● (53, 120)	● (53, 161)
● (54, 137)	● (54, 144)	● (57, 113)	● (57, 168)	● (59, 51)	● (59, 230)	● (61, 84)	● (61, 197)
● (63, 7)	● (63, 274)	● (66, 39)	● (66, 242)	● (68, 114)	● (68, 167)	● (72, 83)	● (72, 198)
● (73, 52)	● (73, 229)	● (74, 9)	● (74, 272)	● (78, 73)	● (78, 208)	● (79, 107)	● (79, 174)
● (80, 79)	● (80, 202)	● (82, 9)	● (82, 272)	● (83, 69)	● (83, 212)	● (87, 126)	● (87, 155)
● (88, 8)	● (88, 273)	● (91, 80)	● (91, 201)	● (93, 74)	● (93, 207)	● (94, 60)	● (94, 221)
● (95, 130)	● (95, 151)	● (96, 66)	● (96, 215)	● (97, 16)	● (97, 265)	● (99, 73)	● (99, 208)
● (100, 78)	● (100, 203)	● (104, 73)	● (104, 208)	● (106, 35)	● (106, 246)	● (107, 113)	● (107, 168)
● (115, 38)	● (115, 243)	● (116, 118)	● (116, 163)	● (117, 113)	● (117, 168)	● (122, 28)	● (122, 253)
● (125, 9)	● (125, 272)	● (126, 11)	● (126, 270)	● (130, 95)	● (130, 186)	● (132, 46)	● (132, 235)
● (136, 81)	● (136, 200)	● (139, 99)	● (139, 182)	● (142, 92)	● (142, 189)	● (145, 78)	● (145, 203)
● (149, 91)	● (149, 190)	● (151, 23)	● (151, 258)	● (155, 21)	● (155, 260)	● (156, 85)	● (156, 196)
● (159, 79)	● (159, 202)	● (164, 88)	● (164, 193)	● (165, 72)	● (165, 209)	● (166, 47)	● (166, 234)
● (174, 88)	● (174, 193)	● (175, 112)	● (175, 169)	● (177, 65)	● (177, 216)	● (181, 81)	● (181, 200)
● (182, 65)	● (182, 216)	● (184, 5)	● (184, 276)	● (185, 126)	● (185, 155)	● (186, 135)	● (186, 146)
● (187, 89)	● (187, 192)	● (188, 12)	● (188, 269)	● (190, 25)	● (190, 256)	● (193, 138)	● (193, 143)
● (194, 66)	● (194, 215)	● (198, 4)	● (198, 277)	● (199, 85)	● (199, 196)	● (201, 28)	● (201, 253)
● (202, 51)	● (202, 230)	● (203, 65)	● (203, 216)	● (207, 85)	● (207, 196)	● (208, 54)	● (208, 227)
● (209, 97)	● (209, 184)	● (213, 140)	● (213, 141)	● (215, 100)	● (215, 181)	● (218, 90)	● (218, 191)
● (220, 44)	● (220, 237)	● (222, 107)	● (222, 174)	● (224, 88)	● (224, 193)	● (227, 45)	● (227, 236)
● (228, 103)	● (228, 178)	● (231, 40)	● (231, 241)	● (234, 102)	● (234, 179)	● (235, 77)	● (235, 204)
● (238, 83)	● (238, 198)	● (239, 28)	● (239, 253)	● (245, 81)	● (245, 200)	● (246, 137)	● (246, 144)
● (247, 104)	● (247, 177)	● (251, 105)	● (251, 176)	● (252, 83)	● (252, 198)	● (253, 98)	● (253, 183)
● (254, 61)	● (254, 220)	● (257, 110)	● (257, 171)	● (261, 107)	● (261, 174)	● (262, 137)	● (262, 144)
● (263, 70)	● (263, 211)	● (264, 14)	● (264, 267)	● (265, 27)	● (265, 254)	● (268, 86)	● (268, 195)
● (272, 66)	● (272, 215)	● (278, 87)	● (278, 194)	● (279, 29)	● (279, 252)	● (280, 132)	● (280, 149)

The results shown here are colour coded dictating the distance the point has from the midpoint which is defined as 250/2. Keeping in mind that for each x value we have two y values, and that the y values range from the midpoint to the modulus function, if we plot the solutions on a two dimensional axis, we will get the spatter that we see in the first image. However, if we plot our equation with the solutions we got on a three dimensional axis, we will get the torus shown next to it. The reason for this is because the equation and its solutions wrap in all directions giving as a result the doughnut shape seen here.





The image displayed above and on the right was created so that the vertical midpoint of the diagram matches with the outer radius of the torus and the top and bottom of the diagram match the inner radius of the torus. This should give you a complete image on how the torus can be represented in a two dimensional form.

As with other forms of cryptography, while using the elliptic curve algorithm certain keys need to be exchanged in order to establish communication and be able to exchange encrypted information and, as a result, to be able to decrypt it for it to be read. Enter Alice and Bob who both use the same elliptic curve and wish to establish a communication between them in order to exchange messages in a secure way. The theory of elliptic curves teaches us that each point (except the control point at infinity) on an elliptic curve can be a prime number generator, thus generate keys suitable for encryption. By freely choosing a point on the curve, Bob and Alice have an infinite number of values to choose from. The bigger the value chosen, the more difficult it will be for the key to be found. So, Alice and Bob both decide on which point on the curve to choose. Let's call this point, I. Each person involved will also choose a secret number to their liking. Let's call these numbers (a) and (b). The next step is for each one of them to replace the point (I) they have chosen with as many times are their respective secret number. This will give Alice a point A = a*I and Bob will have a point B = b*I. Following this, they will exchange their newly found numbers and then multiply the point each one received with their respective secret numbers. By the end of this iteration, Alice will have a point A = a*b*I and Bob will have a point B = b*a*I. This is how the key is generated and distributed to both parties. With this key at hand, the communication is established and ready to commence. While this may seem simple enough and easily breakable, allow me to remind you that both the points and secret numbers are vast prime numbers that can be of 1024 digits long, generating a key of equal size that is the product of two prime numbers of similar length and, as we learned from the trapdoor function earlier, difficult to deduce.

While the method of elliptic curve cryptography is very recent and very secure, with the birth of modern quantum computers it has started to show signs of weakness. A quantum computer can execute a lot of calculations in a fraction of a second and this can pose problems for almost all of the modern cryptography algorithms. In the 1970s, Stephen Wiesner and Gilles Brassard introduced the concept of quantum conjugate coding. Through their work, they showcased how to store or transmit 2 messages by encoding them into conjugate observables, such as linear visualization in a circular polarization of photons. In 1984 Charles H Bennett end Gilles Brassard proposed a method for secure communication which is now called BB84. BB84 was the first quantum cryptographic protocol ever used. Today we are using a method of quantum cryptography that we can safely say that it cannot be hacked.

In order to better understand how quantum encryption works, we must first have a look at Physics and how quantum mechanics come into play when it comes to cryptography. In quantum mechanics there are three principles used: the quantized properties, the particles of light and the wave functions of matter. The first principle describes the quantized properties of the photons such as position, speed and colour that can sometimes only occur in specific set amounts much like a dial that "clicks" (jumps) from number to number. Due to the dual nature of light (this means that it can behave like both as particles (photons; this can be seen by the Brownian motion the particles of dust perform through closed windows in the rays of sunlight) and as waves (as can be seen through Young's experiment with the two slits)), light waves can sometimes cancel each other out (this is called a superposition and can either be constructive (positive superposition) and amplify the signal or destructive (negative superposition) and cancel out the signal all together). So, the bigger the positive superposition, the brighter the light emitted. Wave functions of matter are equations that can describe how matter will behave when it is represented by waves.

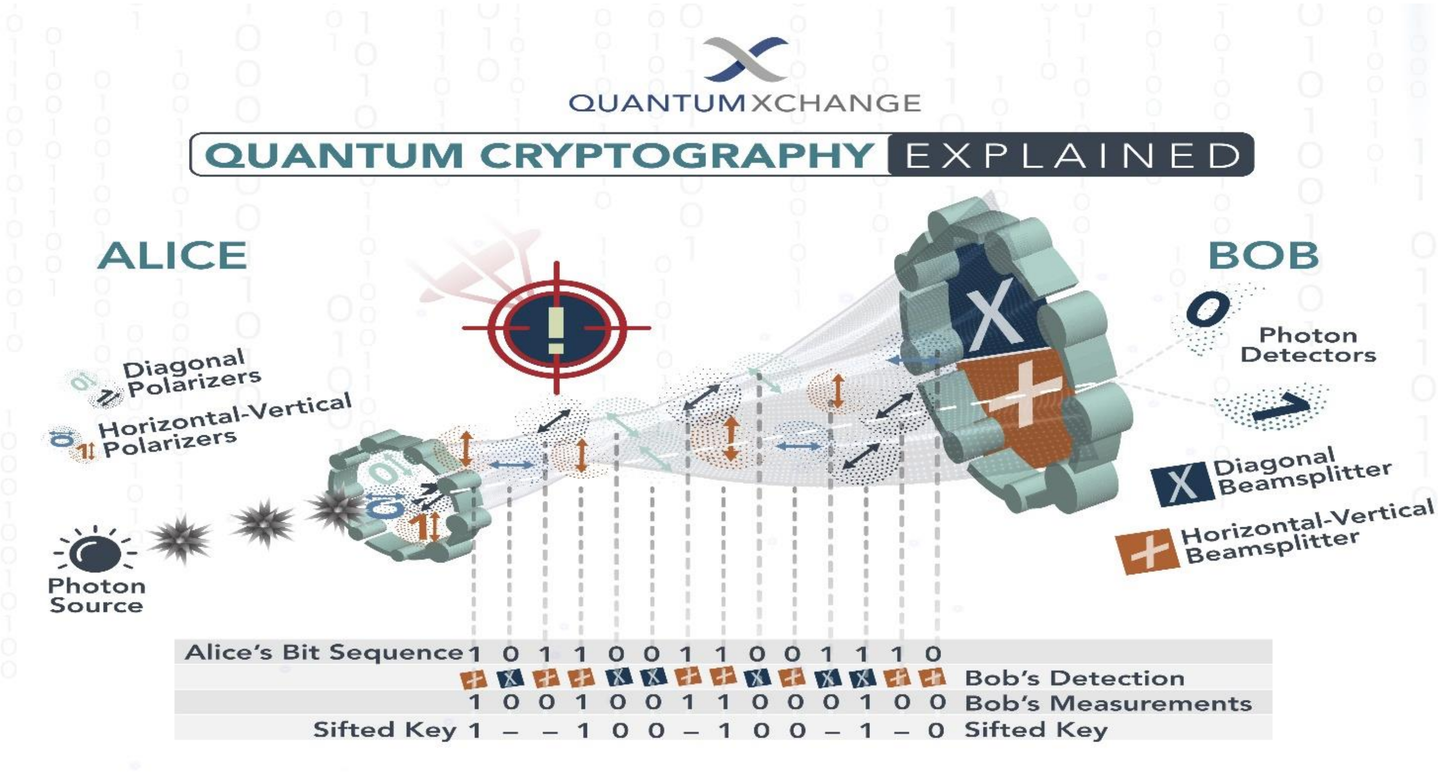
Quantum physics are giving us the basis for the quantum cryptography that we use today. The encryption algorithm for quantum cryptography is based around the following three principles that stem from quantum physics:

1. The sender transmits photons through a filter (or polarizer) which randomly gives them one of four possible polarizations and bit designations: These bits are the Vertical (One bit), the Horizontal (Zero bit), the 45 degree right (One bit), and the 45 degree left (Zero bit). The possible combinations are created between vertical/horizontal and angle on left/right.
2. The photons travel to a receiver, which uses two beam splitters (horizontal/vertical and diagonal) to "read" the polarization of each photon. The receiver does not know which beam splitter to use for each photon and has to guess which one to use.
3. Once the stream of photons has been sent, the receiver tells the sender which beams splitter was used for each of the photons in the sequence they were sent, and the sender compares that information with the sequence of polarizers used to send the key. The photons that were

read using the wrong beam splitter are discarded, and the resulting sequence of bits becomes the key that will be used to encrypt and decrypt the messages in the conversation and data to follow.

Due to the quantum physics principle of observation (based on Heisenberg’s uncertainty principle which states that one can know the velocity or the position of each particle but not both), if an eavesdropper listens in on the conversation and steals the message, they will not be able to determine what the message actually says, no matter how many splitters they will use.

The transmission of information using the method of quantum encryption is done through an optic fibre cable. Due to the sensitive nature of the cable itself, it is rather difficult to gain access to the source of the information (in this case, the light) as it is protected by many layers of material, crucial to its survival. For a better understanding on how quantum cryptography key exchange works, let’s take an example between our favourite couple of Alice and Bob trying to communicate. Alice will send to Bob a series of photons in random states that encode the key to their future conversation. Bob will receive the encrypted key and using the various splitters he will decode it. Let’s assume that an eavesdropper named Eve is trying to listen to their conversation. Despite the importance of information and the frailty of the optic cable, the transmission channel is not even secure. The reason for this is that Eve will have to get the photons, read their state and then pass them on to Bob, so that Bob won’t realize that someone interfered with their conversation. The result of Eve’s interference however is that the quantum state of the photons will change once read, so Bob will alert Alice that the message has been intercepted, thus rendering the first key useless and prompting Alice to send a new key instead.


QUANTUMXCHANGE
QUANTUM CRYPTOGRAPHY EXPLAINED
ALICE
BOB
Diagonal Polarizers
Horizontal-Vertical Polarizers
Photon Source
Photon Detectors
Diagonal Beamsplitter
Horizontal-Vertical Beamsplitter

Alice’s Bit Sequence	1	0	1	1	0	0	1	1	0	0	1	1	1	0	
	+	X	+	+	X	X	+	+	X	+	X	X	+	+	Bob’s Detection
	1	0	0	1	0	0	1	1	0	0	0	1	0	0	Bob’s Measurements
Sifted Key	1	–	–	1	0	0	–	1	0	0	–	1	–	0	Sifted Key

Although quantum cryptography is well established, widely used and virtually unbreakable, with the technology rising, we are bound to find a way to intercept the cryptographic keys at one point, especially with the rise of quantum computers. This has led several scientists to deal with what is now known as post-quantum cryptography. While quantum cryptography is virtually unbreakable, it uses physics properties and through the optic fibre cable (given the right equipment and knowledge) it can be intercepted. The reason it is so secure is because the optic fibre cables are deep in the Earth, covered and concealed, and knowledge in Physics is required to retrieve the information, and at the same time, quantum computers are not readily available and very costly for everyone to get one. In post-quantum cryptography however, along with the principals gained from quantum physics, complex mathematical models and cryptographic algorithms are going to be applied rendering it safer from eavesdroppers and attackers that might use a quantum computer to retrieve the information.

Post quantum computing encryption algorithms are now mainly focused in six different approaches. There are:

1. Lattice based cryptography
2. Multivariate cryptography
3. Hash-based cryptography
4. Code-based cryptography
5. Supersingular elliptic curve isogeny cryptography
6. Symmetric key quantum resistance

Lattice based cryptography was first introduced in 1996 by Miklos Ajtai. His work postulated from the basis that lattice problems are really difficult to solve. Cynthia Dwork proved that a relatively easy lattice problem was as difficult to solve as the worst and most difficult case of lattice problems. Through various calculations, the key produced will be the answer to a known problem of the shortest Euclidean vector value in a lattice with non - zero elements (each time the values in the lattice will differ). This will produce a hash that will be later be used to produce a key that will be transmitted using quantum key exchange.

The multivariate cryptography algorithm produces a key that will be constructed from the solutions of multivariate polynomial equations that are already proven to be of NP-complete difficulty (NP meaning nondeterministic polynomial; its set of solutions can only occur through a brute force algorithm). This will produce a solution that will be hashed and used as the key in quantum cryptography. This form of deriving a key is patented and cannot be used by anyone other than the Rainbow Signature scheme.

The hash-based algorithm is using a digital signature to sign off on every message. This algorithm will produce a series of keys, each one will be used to sign off a message and then it will be discarded. Using this method ensures that even if the keys fall into the wrong hands during one of the exchanges, the eavesdropper will only be able to decrypt a single message from the conversation. This type of hashing algorithms that are used for digital signatures were invented back 1979 by Leslie Lamport. It is worth noting that despite a new key being used every time to encrypt a single message, the hashing algorithm actually uses only one seed key and from that key using various hashing methods all the other keys are produced.

Code-based cryptography is based on error correction codes such as the McEliece cryptosystem. The McEliece cryptosystem uses asymmetric encryption and it was invented by Robert McEliece back in 1978. The algorithm is very powerful and despite that it never caught on, it has a solid reputation as withstanding brute force attacks for the past 30 years. The algorithm produces a key when while trying to decode a linear code (a set of keywords which they are keys themselves in their own merit) a correction error appears. This correction error can be used to generate a key which will be unique. Due to the nature of the decoding of linear code (it is an established NP-hard problem) the key each time is unique, thus ensuring that the key produced for error correction a set of other keys is practically non-decipherable (again, a trapdoor function as we saw before) and results to brute force in case someone wants to obtain it. Needless to say, trying to brute force the solutions to a NP-hard problem with unknown keys is not for the faint hearted, and that is why this algorithm is set to be used as part of the post-quantum cryptography movement.

Perhaps the most intriguing of the proposed solutions for post-quantum cryptography, is the supersingular elliptic curve isogeny cryptography algorithm. This algorithm is based on the Diffie – Hellman algorithm that an unsecure channel is used to transmit a secure key, and that is why it is also known as the supersingular isogeny Diffie–Hellman key exchange, or SIDH. SIDH is designed to resist cryptanalytic attacks by supporting perfect forward secrecy (a technique that ensures that even if the private key of the server is leaked, the session keys between the parties are not compromised at all) and it uses as a walk (path, solutions of the graph) the supersingular isogeny graph. A supersingular isogeny graph is a class of graphs that belong to the expanders, a class of its own that uses vertex, edge or spectral expansion; essentially a graph that is finite and multidirectional with its large solutions not being so large. This property allows the SIDH to boast a small relatively key comprised of 2688 digits and a security level of 128 (meaning that the attacker has to try 2^{128} different combinations of 2688 digit string until it gets the correct one). And since that may not be enough, in order to actually come close to an outcome, the attacker needs to find a perfect solution for isogeny mapping between two supersingular elliptic curves with the same number of points. The high mathematical difficulty, the insane number of possibilities and the combination of so many aspects in terms of theoretical mathematics, random numbers and preset parameters, make this algorithm a perfect candidate for post-quantum cryptography. The algorithm was developed in 2011 by De Feo, Jao, and Plut.

A sparse graph with few edges, part of the family of expanders

Lastly, the algorithm of the symmetric key quantum resistance stems from the AES algorithm that DES uses provided that the keys are large enough numbers. If too big, the keys are already resistant to quantum computing attacks and with the use of protocols like Kerberos which already are resistant to attacks, researches are proposing to replace their current system with this algorithm that will only require an update on the key itself and get post-quantum cryptography today, allowing them to continue to run undisturbed.

The great advancements in cryptography have shown us that with the evolution of the branch, we can definitely see more improvements being made as time goes by. With each new algorithm, with every new discovery we see that the attempts on stealing data and information leaking are diminishing. It is becoming obvious that the combinations of quantum physics, advanced mathematics and old protocols and techniques (like the DH key exchange protocol) are proving to be more efficient in dealing with all the problems modern cryptography faces. The combination of them brings forward advantages from each one of them resulting into a better and safer outcome. Through this study we have seen clearly that what Jacob Appelbaum said about cryptography to be true. No amount of brute force will ever solve a maths problem. I will conclude this study the same way I started it; with another quote, this time by Neil Stephenson, from his book “Cryptonomicon”:

> -How long do you want these messages to remain secret? [...]
>
> -I want them to remain secret for as long as men are capable of evil.

And they will remain a secret. At least, until the next powerful computer comes along…

April 15, 2020, Limassol, Cyprus

References:

- Panayiotis Vryonis, " Explaining public-key cryptography to non-geeks ", August 27, 2013
- Dr. Bill Young, "Foundations of Computer Security Lecture 44: Symmetric vs. Asymmetric Encryption "
- Microsoft, " Description of Symmetric and Asymmetric Encryption "
- Computer Hope," 3DES ", May 22, 2017
- visually, " The History of Encryption", May, 2004
- itchy fish, " Advantages and Disadvantages of Symmetric and Asymmetric Key Encryption Methods "
- Gustavus J. Simmons, "Data Encryption Standard "
- techopedia, " RSA Encryption ", December 4, 2018
- Josh Lake, "What is RSA encryption and how does it work? ", December 10, 2018
- Josh Lake, "What is 3DES encryption and how does DES work? ", February 20, 2019
- John Bailey, " What is elliptic curve cryptography (ECC)? ", November 18, 2016
- Mark Hughes," How Elliptic Curve Cryptography Works ", July 26, 2019
- QUANTUMXCHANGE, " Quantum Cryptography, Explained "
- Robert Coolman, "What Is Quantum Mechanics?", September 26, 2014
- Peter Wilson, “Introduction to Block Ciphers”, 2016
- Brian Curran, “What is RSA Cryptography? Complete Guide to this Encryption Algorithm”, July 6, 2018
- David Kahn, “The codebreakers: the history of secret writing”, 1967
- Joseph H. Silverman, An Introduction to the Theory of Elliptic Curves, June – July, 2006
- Chris Peikert, Lattice Cryptography for the Internet, July 16, 2014
- Tim Guneysu, Vadim Lyubashevsky, and Thomas Poppelmann, “Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems”, 1 Horst Gortz Institute for IT-Security, Ruhr-University Bochum, Germany 2 INRIA / ENS, Paris
- Jiang Zhang, Zhenfeng Zhang, Jintai Ding, Michael Snook, and Ozgur Dagdelen, “Authenticated Key Exchange from Ideal Lattices” 3 1 Institute of Software, Chinese Academy of Sciences, China 2 University of Cincinnati, Cincinnati, USA 3 Darmstadt University of Technology, Germany
- Ding, Jintai; Schmidt, Ioannidis, John, "Rainbow, a New Multivariable Polynomial Signature Scheme", June 7, 2005
- Buchmann, Johannes; Dahmen, Erik; Hülsing, Andreas, "XMSS - A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions" - Post-Quantum Cryptography, 2011
- Bernstein, Daniel J.; Hopwood, Daira; Hülsing, Andreas; Lange, Tanja; Niederhagen, Ruben; Papachristodoulou, Louiza; Schneider, Michael; Schwabe, Peter; Wilcox-O’Hearn, Zooko, Oswald, Elisabeth; Fischlin, Marc (eds.), ”SPHINCS: practical stateless hash-based signatures”, 2015
- Moni Naor, Moti Yung, “Universal One-Way Hash Functions and their Cryptographic Applications”, 1989
- Overbeck, Raphael; Sendrier, Bernstein, Daniel (ed.), ”Code-based cryptography: Post-Quantum Cryptography”, 2009
- Technische Universiteit Eindhoven, “Post-Quantum Cryptography for Long-Term Security”, March 1, 2015
- Sun, Xi; Tian; Wang. Browse Conference Publications > “Intelligent Networking and Co … Help Working with Abstracts Toward Quantum-Resistant Strong Designated Verifier Signature from Isogenies. Intelligent Networking and Collaborative Systems”, (INCoS), 4th International Conference on, September 19 – 22, 2012
- M. Campagna, T. Hardjono, L. Pintsov), B. Romansky and T. Yu, “Kerberos Revisited: Quantum-Safe Authentication”, ETSI Quantum-Safe-Crypto Workshop, September 26, 2013

- Ray A. Perlner, David A. Cooper, “Quantum Resistant Public Key Cryptography: A Survey”, April 14, 2009
- Parker Higgins, “Pushing for Perfect Forward Secrecy, an Important Web Privacy Protection”, August 28, 2013
- Luca De Feo, David Jao, AND Jerome Plut, “Towards quantum – resistant cryptosystems from supersingular elliptic curve isogenies”, May 14, 2014
- Morgan (on Medium), Sparse coding: “A simple exploration”, September 29, 2016
- Neal Stephenson, “Cryptonomicon”, 1999

All images that are included in the study belong to the public domain, have been obtained with the permission of the author or are referenced in the References list and attributed to their respective owners.