CAP Theorem

In theoretical computer science, the CAP theorem, also named Brewer's theorem after computer scientist Eric Brewer, states that it is impossible for a distributed data store to simultaneously provide more than two out of the following three guarantees:

Consistency: Every read receives the most recent write or an error.

Availability: Every request receives a (non-error) response without guarantee that it contains the most recent write

Partition tolerance: The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes

In other words, the CAP theorem states that in the presence of a network partition, one has to choose between consistency and availability. No distributed system is safe from network failures, thus network partitioning generally has to be tolerated. In the presence of a partition, one is then left with two options: consistency or availability. When choosing consistency over availability, the system will return an error or a time-out if particular information cannot be guaranteed to be up to date due to network partitioning. When choosing availability over consistency, the system will always process the query and try to return the most recent available version of the information, even if it cannot guarantee it is up to date due to network partitioning. **In the absence of network failure - that is, when the distributed system is running normally - both availability and consistency can be satisfied**.

Cryptography: its origin

Cryptography, the use of codes and ciphers to protect secrets, began thousands of years ago. Some clay tablets from Mesopotamia somewhat later are clearly meant to protect information - one dated near 1500 BCE was found to encrypt a craftsman's recipe for pottery glaze, presumably commercially valuable. Furthermore, Hebrew scholars made use of simple monoalphabetic substitution ciphers beginning perhaps around 500 to 600 BCE. In India around 400 BCE to 200 CE, the art of understanding writing in cypher, and the writing of words in a peculiar way **was documented in the Kama Sutra for the purpose of communication between lovers.** The ancient Greeks used cryptography. The scytale transposition cipher was used by the Spartan military. **Herodotus tells us of secret messages** physically concealed beneath wax on wooden tablets or **as a tattoo on a slave's head concealed by regrown hair**, although these are not properly examples of cryptography per se as the message, once known, is directly readable; this is known as steganography. The Romans knew cryptography (i.e the famous **Caesar cipher** and its variations).

The modern cryptology originated among the Arabs, the first people to systematically document cryptanalytic methods. The invention of the frequency-analysis technique for breaking monoalphabetic substitution ciphers, by Al-Kindi, an Arab mathematician sometime around AD 800 proved to be the single most significant cryptanalytic advance until World War II. Al-Kindi wrote a book on cryptography entitled Manuscript for the Deciphering Cryptographic Messages, in which he **described the first cryptanalytic techniques, including some for polyalphabetic ciphers, cipher classification, Arabic phonetics and syntax, and most importantly, gave the first descriptions on frequency analysis**. Ahmad al-Qalqashandi (AD 1355–1418) wrote a 14-volume encyclopedia which included a section on cryptology. This information was

attributed to Ibn al-Durayhim who lived from AD 1312 to 1361, but whose writings on cryptography have been lost. **The list of ciphers in this work included both substitution and transposition, and for the first time, a cipher with multiple substitutions for each plaintext letter**. Also traced to Ibn al-Durayhim is an exposition on and worked example of cryptanalysis, including the use of tables of letter frequencies and sets of letters which cannot occur together in one word.

The earliest example of the homophonic substitution cipher is the one used by **Duke of Mantua in the early 1400s.** Homophonic cipher replaces each letter with multiple symbols depending on the letter frequency. The cipher is ahead of the time because it combines monoalphabetic and polyalphabetic features.

Essentially all ciphers remained vulnerable to the cryptanalytic technique of frequency analysis until the development of the polyalphabetic cipher, and many remained so thereafter. The polyalphabetic cipher was most clearly explained by **Leon Battista Alberti around the year AD 1467, for which he was called the "father of Western cryptology". Johannes Trithemius, in his work Poligraphia**, invented the tabula recta, a critical component of the Vigenère cipher. Trithemius also wrote the Steganographia. **The French cryptographer Blaise de Vigenère devised a practical polyalphabetic system** which bears his name, the Vigenère cipher.

In Europe, cryptography became (secretly) more important as a consequence of political competition and religious revolution. For instance, in Europe during and after the Renaissance, citizens of the various Italian states - the Papal States and the Roman Catholic Church included- were responsible for rapid proliferation of cryptographic techniques, few of which reflect understanding (or even knowledge) of Alberti's polyalphabetic advance. 'Advanced ciphers', even after Alberti, weren't as advanced as their inventors / developers / users claimed (and probably even themselves believed). They were regularly broken. This over-optimism may be inherent in cryptography, for it was then - and remains today - fundamentally difficult to accurately know how vulnerable one's system actually is. In the absence of knowledge, guesses and hopes, predictably, are common. **Cryptography, cryptanalysis, and secret-agent/courier betrayal featured in the Babington plot during the reign of Queen Elizabeth I which led to the execution of Mary, Queen of Scots**. Robert Hooke suggested in the chapter Of Dr. Dee's Book of Spirits, that John Dee made use of Trithemian steganography, to conceal his communication with Queen Elizabeth I.

The chief cryptographer of King Louis XIV of France was Antoine Rossignol and he and his family created what is known as the Great Cipher because it remained unsolved from its initial use until 1890, when French military cryptanalyst, Étienne Bazeries solved it. **An encrypted message from the time of the Man in the Iron Mask (decrypted just prior to 1900 by Étienne Bazeries) has shed some, regrettably non-definitive, light on the identity of that real, if legendary and unfortunate, prisoner** .Outside of Europe, after the Mongols brought about the end of the Muslim Golden Age, cryptography remained comparatively undeveloped. Cryptography in Japan seems not to have been used until about 1510, and advanced techniques were not known until after the opening of the country to the West beginning in the 1860s.

Cryptography during the two World Wars

Although cryptography has a long and complex history, it wasn't until the 19th century that it developed anything more than ad hoc approaches to either encryption or **cryptanalysis (the science of finding weaknesses in crypto systems). Examples of the latter include Charles Babbage's Crimean War era work** on mathematical cryptanalysis of polyalphabetic ciphers, redeveloped and published somewhat later by the Prussian **Friedrich Kasiski**. Understanding of cryptography at this time typically consisted of hard-won rules of thumb; see, for example**, Auguste Kerckhoffs'** cryptographic writings in the latter 19th century. **Edgar Allan Poe** used systematic methods to solve ciphers in the 1840s. He later wrote an essay on methods of cryptography which proved useful as an introduction for novice British cryptanalysts attempting to break German codes and ciphers during World War I, and a famous story, The Gold-Bug, in which cryptanalysis was a prominent element.

Cryptography, and its misuse, were involved in the execution of Mata Hari and in Dreyfus' conviction and imprisonment, both in the early 20th century. Cryptographers were also involved in exposing the machinations which had led to the Dreyfus affair; Mata Hari, in contrast, was shot. In World War **I the Admiralty's Room 40 broke German naval codes** and played an important role in several naval engagements during the war, notably in detecting major German sorties into the North Sea that led to the battles of Dogger Bank and Jutland as the British fleet was sent out to intercept them. However its most important contribution was probably in decrypting **the Zimmermann Telegram, a cable from the German Foreign Office sent via Washington to its ambassador Heinrich von Eckardt in Mexico** which played a major part in bringing the United States into the war. In 1917, **Gilbert Vernam** proposed a teleprinter cipher in which a previously prepared key, kept on paper tape, is combined character by character with the plaintext message to produce the cyphertext. This led to the development of electromechanical devices as cipher machines, and to **the only unbreakable cipher, the onetime pad** . During the 1920s, Polish naval-officers assisted the Japanese military with code and cipher development**. Mathematical methods proliferated in the period prior to World War II** (notably in William F. Friedman's application of statistical techniques to cryptanalysis and cipher development and in Marian Rejewski's initial break into the German Army's version of the Enigma system in 1932).

By World War II, mechanical and electromechanical cipher machines were in wide use, although - where such machines were impractical - manual systems continued in use. Great advances were made in both cipher design and cryptanalysis, all in secrecy. Information about this period has begun to be declassified as the official British 50-year secrecy period has come to an end, as US archives have slowly opened, and as assorted memoirs and articles have appeared. **The Germans made heavy use, in several variants, of an electromechanical rotor machine known as Enigma**. Mathematician Marian Rejewski, at Poland's Cipher Bureau, in December 1932 deduced the detailed structure of the German Army Enigma, using mathematics and limited documentation supplied by Captain Gustave Bertrand of French military intelligence. Rejewski and his mathematical Cipher Bureau colleagues, Jerzy Różycki and Henryk Zygalski, continued reading Enigma and keeping pace with the evolution of the German Army machine's components and encipherment procedures. As the Poles' resources became strained by the changes being introduced by the Germans, and as war loomed, the Cipher Bureau, on the Polish General Staff's instructions, on 25 July 1939, at Warsaw, initiated French and British intelligence representatives into the secrets of Enigma decryption.

Soon after the Invasion of Poland by Germany on 1 September 1939, key Cipher Bureau personnel were evacuated southeast ward; on 17 September, as the Soviet Union attacked Poland from the East, they crossed into Romania. From there they reached Paris, France; at PC Bruno, near Paris, they continued breaking Enigma, collaborating with British cryptologists at Bletchley Park as the British got up to speed on breaking Enigma. In due course, **the British cryptographers - whose ranks included many chess masters and mathematics dons such as Gordon Welchman, Max Newman, and Alan Turing** (the conceptual founder of modern computing) - substantially advanced the scale and technology of Enigma decryption. German code breaking in World War II also had some success, most importantly by breaking the Naval Cypher No. 3. This enabled them to track and sink Atlantic convoys. It was only Ultra intelligence that finally persuaded the admiralty to change their codes in June 1943. This is surprising given the success of the British Room 40 code breakers in the previous world war.

At the end of the War, on 19 April 1945, Britain's top military officers were told that they could never reveal that the German Enigma cipher had been broken because it would give the defeated enemy the chance to say they "were not well and fairly beaten". US Navy cryptographers (with cooperation from British and Dutch cryptographers after 1940) broke into several Japanese Navy crypto systems. **The break into one of them, JN-25, famously led to the US victory in the Battle of Midway**; and to the publication of that fact in the Chicago Tribune shortly after the battle, though the Japanese seem not to have noticed for they kept using the JN-25 system. A US Army group, the SIS, managed to break the highest security Japanese diplomatic cipher system (an electromechanical 'stepping switch' machine called Purple by the Americans) even before World War II began. The Americans referred to the intelligence resulting from cryptanalysis, perhaps especially that from the Purple machine, as 'Magic'. The British eventually settled on 'Ultra' for intelligence

resulting from cryptanalysis, particularly that from message traffic protected by the various Enigmas. An earlier British term for Ultra had been 'Boniface' in an attempt to suggest, if betrayed, that it might have an individual agent as a source.

The German military also deployed several mechanical attempts at a one-time pad. Bletchley Park called them the Fish ciphers, and Max Newman and colleagues designed and deployed the Heath Robinson, and then the world's first programmable digital electronic computer, the Colossus, to help with their cryptanalysis. The German Foreign Office began to use the one-time pad in 1919; some of this traffic was read in World War II partly as the result of recovery of some key material in South America that was discarded without sufficient care by a German courier. The Japanese Foreign Office used a locally developed electrical stepping switch based system (called Purple by the US), and also had used several similar machines for attaches in some Japanese embassies. One of the electrical stepping switch based systems referred to earlier as Purple was called the 'M-machine' by the US, another was referred to as 'Red'. All were broken, to one degree or another, by the Allies. Allied cipher machines used in World War II included the British TypeX and the American SIGABA; both were electromechanical rotor designs similar in spirit to the Enigma, albeit with major improvements. Neither is known to have been broken by anyone during the War. The Poles used the Lacidamachine, but its security was found to be less than intended (by Polish Army cryptographers in the UK), and its use was discontinued. US troops in the field used the M-209 and the still less secure M-94 family machines. **British SOE agents initially used 'poem ciphers' (memorized poems were the encryption/decryption keys)**, but later in the War, they began to switch to one-time pads. The VIC cipher (used at least until 1957 in connection with Rudolf Abel's NY spy ring) was a very complex hand cipher, and is claimed to be the most complicated known to have been used by the Soviets.

Modern cryptography

Claude E. Shannon is considered by many to be the father of mathematical cryptography. Shannon worked for several years at Bell Labs, and during his time there, he produced an article **entitled "A mathematical theory of cryptography".** This article was written in 1945 and eventually was published in the Bell System Technical Journal in 1949. It is commonly accepted that this paper was the starting point for development of modern cryptography. Shannon was inspired during the war to address the problems of cryptography because secrecy systems furnish an interesting application of communication theory. **Shannon identified the two main goals of cryptography: secrecy and authenticity**. His focus was on exploring secrecy and thirty-five years later, G.J. Simmons would address the issue of authenticity. Shannon wrote a further article entitled "A mathematical theory of communication" which highlights one of the most significant aspects of his work: **cryptography's transition from art to science**. In his works, Shannon described **the two basic types of systems for secrecy. The first are those designed with the intent to protect against hackers and attackers who have infinite resources with which to decode a message** (theoretical secrecy, now unconditional security), and the **second are those designed to protect against hackers and attacks with finite resources** with which to decode a message (practical secrecy, now computational security).

Most of Shannon's work focused around theoretical secrecy; here, Shannon introduced a definition for the **"unbreakability"** of a cipher. If a cipher was determined "unbreakable", it was considered to have "perfect secrecy". In proving "perfect secrecy", Shannon determined **that this could only be obtained with a secret key whose length given in binary digits was greater than or equal to the number of bits contained in the information being encrypted.** Furthermore, Shannon developed the "unicity distance", defined as the "amount of plaintext that... determines the secret key." Shannon's work influenced further cryptography research in the 1970s, as the public-key cryptography developers, M. E. Hellman and W. Diffie cited Shannon's research as a major influence. His work also impacted modern designs of secret-key ciphers. At the end of Shannon's work with cryptography, progress slowed until Hellman and Diffie introduced their paper involving "public-key cryptography".

Until the 1970s, secure cryptography was largely the preserve of governments. Encryption in modern times is achieved by using algorithms that have a key to encrypt and decrypt information. These keys convert the messages and data into "digital gibberish" through encryption and then return them to the original form through decryption. In general, the longer the key is, the more difficult it is to crack the code. This holds true because deciphering an encrypted message by brute force would require the attacker to try every possible key. To put this in context, each binary unit of information, or bit, has a value of 0 or 1. An 8-bit key would then have 256 or 2^8 possible keys. A 56-bit key would have 2^{56}, or 72 quadrillion, possible keys to try and decipher the message. With modern technology, cyphers using keys with these lengths are becoming easier to decipher. DES, an early US Government approved cypher, has an effective key length of 56 bits, and test messages using that cypher have been broken by brute force key search. However, as technology advances, so does the quality of encryption**. Since World War II, one of the most notable advances in the study of cryptography is the introduction of the asymmetric key cyphers** (sometimes termed public-key cyphers). These are algorithms which use two mathematically related keys for encryption of the same message. Some of these algorithms permit publication of one of the keys, due to it being extremely difficult to determine one key simply from knowledge of the other.

IV LESSON: MODERN CRIPTOGRAPHY AND ELLIPTIC CURVES

The mid-1970s saw two major public (i.e., non-secret) advances. First was the publication of the draft Data Encryption Standard in the U.S. Federal Register on 17 March 1975. The proposed DES cipher was submitted by a research group at IBM, at the invitation of the National Bureau of Standards (now NIST), in an effort to develop secure electronic communication facilities for businesses such as banks and other large financial organizations. After advice and modification by the NSA, acting behind the scenes, it was adopted and published as a Federal Information Processing Standard Publication in 1977 (currently at FIPS 46-3). **DES was the first publicly accessible cipher to be 'blessed' by a national agency such as the NSA.** The release of its specification by NBS stimulated an explosion of public and academic interest in cryptography. **The aging DES was officially replaced by the Advanced Encryption Standard (AES) in 2001** when NIST announced FIPS 197. After an open competition**, NIST selected Rijndael, submitted by two Belgian cryptographers**, to be the AES. DES, and more secure variants of it (such as Triple DES), are still used today, having been incorporated into many national and organizational standards. However, its 56-bit key-size has been shown to be insufficient to guard against brute force attacks (one such attack, undertaken by the cyber civil-rights group Electronic Frontier Foundation in 1997, succeeded in 56 hours.) As a result, use of straight DES encryption is now without doubt insecure for use in new cryptosystem designs, and **messages protected by older cryptosystems using DES, and indeed all messages sent since 1976 using DES, are also at risk**. Regardless of DES' inherent quality, the DES key size (56-bits) was thought to be too small by some even in 1976, perhaps most publicly by Whitfield Diffie. There was suspicion that government organizations even then had sufficient computing power to break DES messages; clearly others have achieved this capability.

The second development, in 1976, was perhaps even more important, for it fundamentally changed the way cryptosystems might work. This **was the publication of the paper New Directions in Cryptography by Whitfield Diffie and Martin Hellman**. It introduced a radically new method of distributing cryptographic keys, which went far toward solving one of the fundamental problems of cryptography, key distribution, and has become known as Diffie - Hellman key exchange. The article also stimulated the almost immediate public development of a new class of enciphering algorithms, the asymmetric key algorithms. **Prior to that time, all useful modern encryption algorithms had been symmetric key algorithms, in which the same cryptographic key is used with the underlying algorithm by both the sender and the recipient, who must both keep it secret.** All of the electromechanical machines used in World War II were of this logical class, as were the Caesar ciphers and essentially all cipher systems throughout history. The 'key' for a code is, of course, the codebook, which must likewise be distributed and kept secret, and so shares most of the same problems in practice.

Of necessity, **the key in every such system had to be exchanged between the communicating parties in some secure way prior to any use of the system** (the term usually used is 'via a secure channel') such as a **trustworthy courier with a briefcase handcuffed to a wrist, or face-to-face contact, or a loyal carrier pigeon**. This requirement is never trivial and very rapidly becomes unmanageable as the number of participants increases, or when secure channels aren't available for key exchange, or when, as is sensible cryptographic practice, keys are frequently changed. In particular, if messages are meant to be secure from other users, a separate key is required for each possible pair of users. A system of this kind is known as a secret key, or symmetric key cryptosystem. **D-H key exchange (and succeeding improvements and variants) made operation of these systems much easier, and more secure, than had ever been possible before in all of history**. In contrast, asymmetric key encryption uses a pair of mathematically related keys, each of which decrypts the encryption performed using the other. **Some, but not all, of these algorithms have the additional property that one of the paired keys cannot be deduced from the other by any known method other than trial and error. An algorithm of this kind is known as a public key or asymmetric key system**. Using such an algorithm, only one key pair is needed per user. By designating one key of the pair as private (always secret), and the other as public (often widely available), no secure channel is needed for key exchange. So long as the private key stays secret, the public key can be widely known for a very long time without compromising security, making it safe to reuse the same key pair indefinitely.

For two users of an asymmetric key algorithm to communicate securely over an insecure channel, each user will need to know their own public and private keys as well as the other user's public key. Take this basic scenario**: Alice and Bob each have a pair of keys they've been using for years with many other users. At the start of their message, they exchange public keys, unencrypted over an insecure line. Alice then encrypts a message using her private key, and then re-encrypts that result using Bob's public key.** The double-encrypted message is then sent as digital data over a wire from Alice to Bob. Bob receives the bit stream and decrypts it using his own private key, and then decrypts that bit stream using Alice's public key. If the final result is recognizable as a message, Bob can be confident that the message actually came from someone who knows Alice's private key (presumably actually her if she's been careful with her private key), and that anyone eavesdropping on the channel will need Bob's private key in order to understand the message.

Asymmetric algorithms rely for their effectiveness on a class of problems in mathematics called one-way functions, which require relatively little computational power to execute, but vast amounts of power to reverse, if reversal is possible at all. A classic example of a one-way function is multiplication of very large prime numbers. It's fairly quick to multiply two large primes, but very difficult to find the factors of the product of two large primes. Because of the mathematics of one-way functions, most possible keys are bad choices as cryptographic keys; only a small fraction of the possible keys of a given length are suitable, and so asymmetric algorithms require very long keys to reach the same level of security provided by relatively shorter symmetric keys. The need to both generate the key pairs, and perform the encryption/decryption operations make asymmetric algorithms computationally expensive, compared to most symmetric algorithms. Since symmetric algorithms can often use any sequence of (random, or at least unpredictable) bits as a key, a disposable session key can be quickly generated for short-term use. **Consequently, it is common practice to use a long asymmetric key to exchange a disposable, much shorter (but just as strong) symmetric key. The slower asymmetric algorithm securely sends a symmetric session key, and the faster symmetric algorithm takes over for the remainder of the message.**

Asymmetric key cryptography, Diffie - Hellman key exchange, and the best known of the public key / private key algorithms (i.e., what is usually called the RSA algorithm), all seem to have been independently developed at a UK intelligence agency before the public announcement by Diffie and Hellman in 1976. GCHQ has released documents claiming they had developed public key cryptography before the publication of Diffie and Hellman's paper. Various classified papers were written at GCHQ during the 1960s and 1970s which eventually led to schemes essentially identical to RSA encryption and to Diffie - Hellman key exchange in 1973 and 1974. Some of these have now been published, and the inventors (James H. Ellis, Clifford Cocks, and Malcolm Williamson) have made public (perhaps, some of) their work. **Beginning around 1990, the use of the Internet for commercial purposes and the introduction of commercial transactions over the Internet**

called for a widespread standard for encryption. Before the introduction of the Advanced Encryption Standard (AES), information sent over the Internet, such as financial data, was encrypted if at all, most commonly using the Data Encryption Standard (DES).. Around the late 1990s to early 2000s, the use of public-key algorithms became a more common approach for encryption, and soon a hybrid of the two schemes became the most accepted way for e-commerce operations to proceed. **Additionally, the creation of a new protocol known as the Secure Socket Layer, or SSL, led the way for online transactions to take place. Transactions ranging from purchasing goods to online bill pay and banking used SSL. Furthermore, as wireless Internet connections became more common among households, the need for encryption grew, as a level of security was needed in these everyday situations.**

As we will say also later, when we will describe how bitcoins works, **hashing is a common technique** used in cryptography to encode information quickly using typical algorithms. **Generally, an algorithm is applied to a string of text, and the resulting string becomes the "hash value".** This creates a "digital fingerprint" of the message, as the specific hash value is used to identify a specific message**. The output from the algorithm is also referred to as a "message digest" or a "check sum".** Hashing is good for determining if information has been changed in transmission. If the hash value is different upon reception than upon sending, there is evidence the message has been altered. Once the algorithm has been applied to the data to be hashed, the hash function produces a fixed-length output. Essentially, anything passed through the hash function should resolve to the same length output as anything else passed through the same hash function. It is important to note that hashing is not the same as encrypting. **Hashing is a one-way operation that is used to transform data into the compressed message digest. Additionally, the integrity of the message can be measured with hashing.** Conversely, encryption is a two-way operation that is used to transform plaintext into cipher-text and then vice versa. In encryption, the confidentiality of a message is guaranteed

Hash functions can be used to verify digital signatures, so that when signing documents via the Internet, the signature is applied to one particular individual. Much like a hand-written signature, these signatures are verified by assigning their exact hash code to a person. Furthermore, hashing is applied to passwords for computer systems. **Hashing for passwords began with the UNIX operating system**. A user on the system would first create a password. That password would be hashed, using an algorithm or key, and then stored in a password file. This is still prominent today, as web applications that require passwords will often hash user's passwords and store them in a database. As a natural consequence, **the public developments of the 1970s broke the near monopoly on high quality cryptography held by government organizations. For the first time ever, those outside government organizations had access to cryptography not readily breakable by anyone (including governments).** Considerable controversy, and conflict, both public and private, began more or less immediately, sometimes called the crypto wars. They have not yet subsided. In many countries, for example, export of cryptography is subject to restrictions. Until 1996 export from the U.S. of cryptography using keys longer than 40 bits (too small to be very secure against a knowledgeable attacker) was sharply limited. **As recently as 2004, former FBI Director Louis Freeh, testifying before the 9/11 Commission, called for new laws against public use of encryption**.

One of the most significant people favouring strong encryption for public use was Phil Zimmermann. He wrote and then in 1991 released PGP (Pretty Good Privacy), a very high quality crypto system. He distributed a freeware version of PGP when he felt threatened by legislation then under consideration by the US Government that would require backdoors to be included in all cryptographic products developed within the US. His system was released worldwide shortly after he released it in the US, and that began a long criminal investigation of him by the US Government Justice Department for the alleged violation of export restrictions. The Justice Department eventually dropped its case against Zimmermann, and the freeware distribution of PGP has continued around the world. PGP even eventually became an open Internet standard . While modern ciphers like AES and the higher quality asymmetric ciphers are widely considered unbreakable, poor designs and implementations are still sometimes adopted and there have been important cryptanalytic breaks of deployed crypto systems in recent years. Notable examples of broken crypto designs include the first Wi-Fi encryption scheme WEP, the Content Scrambling System used for encrypting and controlling DVD use, the A5/1 and A5/2 ciphers used in GSM cell phones, and the CRYPTO1 cipher used in the widely deployed MIFARE

Classic smart cards from NXP Semiconductors, a spun off division of Philips Electronics. All of these are symmetric ciphers. **Thus far, not one of the mathematical ideas underlying public key cryptography has been proven to be 'unbreakable', and so some future mathematical analysis advance might render systems relying on them insecure. While few informed observers foresee such a breakthrough, the key size recommended for security as best practice keeps increasing as increased computing power required for breaking codes becomes cheaper and more available**. In the sequel, we consider cryptographic systems based on elliptic curves that are the one used by bitcoins.

Elliptic curves

Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-ECC cryptography (based on plain Galois fields) to provide equivalent security. Elliptic curves are applicable for key agreement, digital signatures, pseudo-random generators and other tasks. Indirectly, they can be used for encryption by combining the key agreement with a symmetric encryption scheme**. Public-key cryptography is based on the intractability of certain mathematical problems. Early public-key systems are secure assuming that it is difficult to factor a large integer composed of two or more large prime factors. For elliptic-curve-based protocols, it is assumed that finding the discrete logarithm of a random elliptic curve** element with respect to a publicly known base point is infeasible: this is the "elliptic curve discrete logarithm problem" (ECDLP). The security of elliptic curve cryptography depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original and product points. The size of the elliptic curve determines the difficulty of the problem.

For current cryptographic purposes, **an elliptic curve** is a plane curve over a finite field (rather than the real numbers) which consists of the points **satisfying the equation**

$y^2 = x^3 + ax + b,$

along with a distinguished point at infinity. In the mathematics of the real numbers, the logarithm $\log_b a$ is a number x such that $b^x = a$, for given numbers a and b. Analogously, in any group G, powers b^k can be defined for all integers k, and the discrete logarithm $\log_b a$ is an integer k such that $b^k = a$**. The use of elliptic curves in cryptography was suggested independently by Neal Koblitz and Victor S. Miller in 1985. Elliptic curve cryptography algorithms entered wide use in 2004 to 2005.**

The curve used by Bitcoin, secp256k1, in the normal Weierstrass form has equation $y^2 = x^3 + 7$. The elliptic curve can take characteristic shapes in the plane according to its coefficients, but each one is symmetrical with respect to the abscissa axis, since for each value of x there will be a positive and a negative value for y, that is: $y = \pm(x^3 + ax + b)^{(1/2)}$. In cryptography, curves are used on which some algebraic properties can be defined with respect to an internal composition operation, therefore **only non-singular curves will be taken into consideration, discarding all those curves with cusps or with self-intersections**

To verify the non-singularity of the curve, it is necessary to impose that its determinant is different from 0, i.e. that the inequality exists: $4a^3 + 27b^2$ different from 0 The points of a non-singular curve, combined with a special element 0 called point to infinity or zero point, represent a set G, defined in this way:

$G = \{(x,y) \in R^2 \mid y^2 = x^3 + ax + b,\ 4a^3 + 27b^2 \text{ different from } 0\} \cup \{0\}$

A commutative, or abelian, group is a non-empty set on which a · binary operation is defined to satisfy certain properties:

1. the set is closed with respect to the operation, i.e. if a and b belong to the set G then also c = a·b belongs to G;

2. the operation respects the associative property, or (a·b)·c = a·(b·c);

3. there is a 0 element, called identity element, such that a·0 = a and 0·a = a for every a;

4. each element has its inverse, that is, for every a, there exists b such that a·b = 0;

5. the operation respects the commutative property, or a·b = b·a for each a and b belonging to the set.

V LESSON: MORE ABOUT ELLIPTIC CURVES. AN INTRODUCTION TO QUANTUM COMPUTING

A group that contains a finite number of elements is called a finite group and the number of elements in the group is the group order, otherwise the group is called an infinite group. On a G group you can define the operation of elevation to power as the repeated application of the group operator, so $a^3 = a \cdot a \cdot a$. A G group is called cyclic if each element of G is a power a^k of a fixed element $a \in G$, with $k \in N$, in this case it is he says that the element a generates the group G or that is a generator of G, moreover a cyclic group is always abelian and can be finite or infinite. In the case of elliptic curves, the composition operation is the sum, indicated with the symbol +.

Moreover:

- the **inverse** of a point $P(x_P,y_P)$ is defined as the point $-P(x_P,-y_P)$ symmetric of P with respect to the axis x;

- the **identity element** is represented by the point to infinity, or zero point 0 for which is worth 0 = −0 and for every point P belonging to G we have P + 0 = 0 + P = P;

- the **sum operation**, indicated with + is defined by the rule P + Q + R = 0, with P,Q and R belonging to the set G and is aligned.

The elements of the group can be represented as points on the Cartesian plane and also the law of internal composition can be interpreted in a geometric way, establishing that if three points of the curve lie on the same line, or are aligned, their sum is zero. As we have to do with abelian group, it is guaranteed that each element has an inverse element with respect to the sum and that the operation of sum has the commutative property, so that the rule for the sum can be rewritten as P + Q = −R , where P,Q and R are aligned points. To calculate the sum between two points P and Q belonging to the curve we must draw a straight line between them until you find a third intersection point R, the result of the sum will be the inverse of the point of intersection −R, symmetric of R with respect to the x axis.

Depending on the combination between the line passing through the two points and the curve, **we can define the sum in various way:**

1. if one of the two addends is the zero point, we cannot draw any line, since 0 does not belong to the curve, but by definition of element identity we can write P + 0 = P;

2. If one of the two addends is the inverse of the other, then the line passing between the two addends is a vertical line and does not intersect the curve at any point, but by definition of the inverse element we will have $P + (-P) = 0$;

3. if the two addends P and Q coincide, then $P = Q$ and for them pass infinite lines, among these we choose the tangent line to the curve passing through the point P, until you find the intersection R with the curve, at this point we will have $P + P = -R$;

The algebraic treatment of the sum proceeds in a similar way with respect the geometric version with various possible cases. The cases for which the result descends directly from the definition of element identity $P + 0 = P$, and inverse element that is $P + (-P) = 0$, are trivial. In the case where P and Q are two non-symmetrical, non-null and distinct points, we have that the line that unites them has an angular coefficient:

$m = (y_P - y_Q)/(x_P - x_Q)$

The point of intersection between the line passing through the two points and the elliptic curve is a third point $R = (x_R, y_R)$ whose coordinates are defined in this way: $x_R = m^2 - xP - xQ$, $y_R = y_P + m \times (x_R - x_P)$

That is $P + Q = -R$ where $-R = (x_R, -y_R)$.

In the particular case where **we want to define the sum P + P we have to use the tangent to the curve** in point P and it is necessary to use the formula of the first derivative with respect to x of the curve equation:

$m = (3x^2_P + a)/(2y_P)$

We have seen that the set of points belonging to an elliptic curve, with the addition of the point to infinity, constitutes, with respect to the previously defined sum operation, an abelian group. We have formalized the details of the summing operation between two points belonging to the curve, this allows us to define a scalar multiplication operation of a point P belonging to the curve, for a natural number: $nP = P + P \cdots + P$ for n times. **The multiplication of an element of the group for a scalar, that is the repeated application of the sum operator**, by definition represents the elevation to power within the group G, or $P^3 = 3P = P + P + P$ and **the inverse of this operation will be called logarithm on elliptic curves.**

Based on the algebraic formulas introduced previously for the sum of two points, we can perform the previous multiplication by making n−1 sum operations, actually, with the use of appropriate algorithms we can do much better . **One of the algorithms that can be used to efficiently implement the scalar multiplication operation is the double and add algorithm.** Given the product $n*P$, with $n \in N$ and $P \in G$, a generic scalar n can be written as the sum $n_0 + 2n_1 + 2^2n_2 \cdots + 2^m n_m$, where $[n_0, \cdots, n_m] \in \{0,1\}$ and $m + 1$ is the number of digits of the binary representation of n. Suppose we want to multiply the generic point P for 151, whose binary representation is 100101112, then we can write:

$151*P = 2^7P + 2^4P + 2^2P + 2^1P + 2^0P$

The double and add algorithm tells us:

initialize the result Q to 0;

with $i = 0$, since $d_0 = 1$ we add P to Q and store the result in Q and double P;

with i = 1, since $d_1 = 1$ we add P to Q and store the result in Q and double P;

with i = 2, since $d_2 = 1$ we add P to Q and store the result in Q and double P;

with i = 3, since $d_3 = 0$ we do not execute any sum, but we double P;

with i = 4, since $d_4 = 1$ we add P to Q and store the result in Q and double P;

with i = 5, since $d_5 = 0$ we do not execute any sum, but we double P;

with i = 6, since $d_6 = 0$ we do not execute any sum, but we double P;

with i = 7, since $d_7 = 1$ we add P to Q and store the result in Q and double P;

no binary digits of n are left to be taken into account, then returns Q.

The algorithm gives the result of multiplication by executing 5 sums and 7 multiplications. For each iteration of the loop this algorithm performs a summing operation, or alternatively a summing operation followed by another summing operation (doubling P), the loop is executed as many times as the binary digits of n, this leads us to **estimate a cost of O(logn).** So far we have talked about elliptic curves in which the variables and the coefficients belong to the real numbers, but **in their cryptographic application both the variables and the coefficients are restricted to the elements of a finite field**. In mathematics, a finite field, or Galois field, is a field with a finite number p^n of elements, with p prime number and is often denoted as $Z(p^{n)}$ or $GF(p^n)$.

The security of elliptic curve cryptography depends on the difficulty with which it is possible to perform the inverse operation, i.e. to determine n when n*P and P are given. This problem is called the discrete logarithm of the elliptic curve and it is a problem that is considered hard. **Currently the fastest known technique for calculating the logarithm is called the Pollard rho method. Designed by John Pollard in 1975, it was used in 1981 to factor Fermat's eighth number issue** (a Fermat number, named after Pierre de Fermat who first studied them, is a positive integer of the form $F_n = (2^2)^n + 1$. It was conjectured that all the Fermat number were prime number, conjecture that was proved to be false) **It is a probabilistic algorithm, in the sense that it does not guarantee to produce a result.**

In reality **there are some elliptic curves for which it is possible to find specific algorithms that solve the discrete logarithm in polynomial time**, such curves are not suitable for cryptographic uses and are therefore called weak. The possibility that some curves are intrinsically weak to a cryptographic analysis imposes several questions related to the trust that it is legitimate to place in objects of this type. Suppose, in fact, that someone proposes the use of a curve, how can we be sure that it does not have some kind of mathematical vulnerability not yet discovered that makes the problem of the logarithm solvable in polynomial times ? **To avoid** the eventuality that some attacker can forge a curve so as to include in it some **mathematical back-doors it is used the principle called nothing up my sleeve**, that is it is introduced a random number, called seed, which is used to generate curve parameters and the generator point, using hash functions. A curve generated by the use of a seed is called verifiably random, or randomly verifiable

Quantum Computing: the next (?) revolution in cryptography.

Quantum computing uses quantum-mechanical phenomena, such as **superposition and entanglement.**

We recall that **quantum superposition** is a fundamental principle of quantum mechanics. It states that, much like waves in classical physics, any two (or more) quantum states can be added together ("superposed")