POST QUANTUM CRYPTOGRAPHY
A brief survey of classical cryptosystems, their fallacy and the advent of post-quantum cryptography with the deep insight into hashed based signature scheme

Sawan Bhattacharyya[1] and **Dr.Amlan Chakrabarti**[2]

[1]Department of Computer Science,Ramakrishna Mission Vivekananda Centenary College, Kolkata,West Bengal,India
[2]A.K.Choudhury School of Information Technology,University of Calcutta, Kolkata,West Bengal,India
[1]+91-8918685458
[2]+91-9831129520
[1]sawanbhattacharyyakv@gmail.com
[2]acakcs@caluniv.ac.in

Abstract

Cryptography is the art of writing secret text. Employing Cryptography the human-readable text is converted into a Ciphertext(coded text) by the means of some consistent algorithm. The essence of a good Cryptosystem lies in consistent and easy to understand Encryption and Decryption algorithms. Before the advent of Quantum computers, much of the cryptographic algorithms depend on the principles of number theory, abstract algebra, and probability theory mainly Factorization Problem and Discrete Logarithmic problems. With the advancements in the field of quantum computing, the classical cryptosystems are no longer secure and it poses a threat to global security. In this brief survey, we would look towards some of the prime classical cryptosystems, their fallacy, and the advancement of the post-quantum cryptography, especially of the Hashed based Signature Scheme. The review also looks towards the difference between Post Quantum Cryptography and Quantum Cryptography.

Keywords: Classical Cryptosystem,Hash Functions,Post Quantum Cryptography,Private Key Cryptography,Public-Key Cryptosystem, Qubits,Quantum Entanglement,Quantum Fourier Transform,Quantum Key Distribution,Quantum Superposition,

1 Introduction

Privacy had been an indispensable part of human civilization. From the very initial phases of communication, humans have focused on privacy to share valuable information and facts. The prime objective had not changed since then and it focuses on that no one except those directly being involved over communication can get accessed to the information. Modern Cryptosystems makes it possible to communicate even over some insecure channels. The first known evidence of the use of cryptography can be traced backed to 1900 BC in an inscription carved in the main chamber of the tomb of the nobleman Khnumhotep 2 in Egypt.
The human-readable messages are called plaintext. The plaintext is encrypted using some consistent algorithm,(encryption algorithm) into a coded text known as Ciphertext using some keys which can either be private or public. The encoded ciphertext is decrypted using some consistent algorithm,(decryption algorithm) back into the original human-readable plaintext by using either the same keys used during the encryption algorithm or some other

keys. The keys must be safely exchanged between the parties involved in the active communication using some protocols, most famous was, "*New direction in cryptography, IEEE,1976*" widely known as Diffie-Hellman Key Exchange Protocol. It's worthwhile to mention that both the encryption and decryption algorithms are public, the only secret is the key and thus the need for secure key exchange protocol is clear.
There are three cryptographic techniques based on the key and encryption

1. **Private Key Cryptosystems:**Presence of a single secret shared key between the two parties.

2. **Public Key Cryptosystems:**Presence of two keys one is a secret key and the other is public.

3. **Hash Function:**Presence of one way cryptographic Hash-Functions

Before the development of Quantum Computer, much of the cryptographic algorithms were done based on the knowledge of Number theory, Abstract Algebra and the security issues with all pre-Quantum Cryptographic era relies on two main mathematical principles

1. Factoring Problem:Factoring the product of two large primes

2. Discrete Logarithmic Problem:Given $A = g^b \bmod p$, and finding the value of b where A came from some group and g is its generator

But in the year of 1994, Peter Shor of Bell Laboratories published his paper named, *"Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer"* popularly known as the Shor's Algorithm [5] showed that Quantum Computer using the principle of Quantum Superposition and Quantum Entanglement can solve the above two mathematical problems. [14] All of our classical cryptosystem's security is based on the hardness of the two problems but now they can easily be solved even when the number involved is quite large. Thus quantum computers put our pre-existing cryptosystems and consequently our global security is at a risk.
These days Private Key Encryption Key is no longer used as Key Exchange is a major problem and if the communication network has n users then there is a need for $\binom{n}{2}$ secret keys which are not a feasible solution when n is large hence private key encryption is obsolete nowadays and Quantum Computer seems to have a very little effect on Private Key Encryption methods AES(Advanced Encryption Standard) believed to be safe against quantum attacks by choosing somewhat larger keys but no one had ever applied Shor Algorithm to AES. There also exists another quantum algorithm named Grover's algorithm but it is not as shockingly fast as Shor's algorithm and cryptographer can easily compensate for it by choosing a somewhat larger key size.
Grover's algorithm provides a quadratic speed up for the quantum search algorithm in comparison to the search algorithm available on the classical computer, the practical relevancy of Grover's Algorithm is unknown as Shor' Algorithm but if it is practically relevant doubling key size will be fine.
Furthermore, it has been seen that exponential speedup for the quantum search algorithm is impossible [5] and it thus suggests that symmetric key algorithms and hash-based cryptography can be safely used in the quantum era.
The security issues with the public key cryptosystem rely on the use of trapdoor one-way function as proposed by Diffie and Hellman in the year 1976 such that the information of one thing does not exploit the privacy of the other thing unless we are equipped with some additional information in another word we need a function that is easy to compute in one way (anybody can encrypt the message) and that is hard to invert unless we have any additional information called a trapdoor. Therefore, these functions are called trapdoor one-way functions. Now, what, all our pre-existing classical cryptosystems had been dead and there is a need for some new cryptosystems which would be resistant against both quantum and classical attack. [15]

These four classes of cryptosystem are quite promising

1. **Hash-based Signature Scheme**

2. **Lattice-based Cryptosystem**

3. **Multivariate Cryptosystem**

4. **Code-based Cryptosystem**

Apart from these four classes, few more cryptosystems are resistant to quantum attack, one such cryptosystem involves the evaluation of the isogenies of the supersingular elliptic curve. [6] The discrete logarithm problem on an elliptic curve can easily solve using a quantum computer but not the isogeny problem on a supersingular elliptic curve.

2 Litarature Review

2.1 Classical Cryptosystems

The classical cryptosystems are primarily based on the principle of factorization and discrete logarithmic problems. There are three cryptographic techniques based on the key and encryption

1. **Private Key Cryptosystems:**
   In Private Key Cryptosystems, there is one single key which is used both in the encryption as well as decryption process and is kept secret between the receiver and sender, also called Symmetric Key cryptography. Example - Caesar cipher, Substitution cipher, Play fair cipher, Vigenere cipher, AES-128,192,256(refer Figure.1).

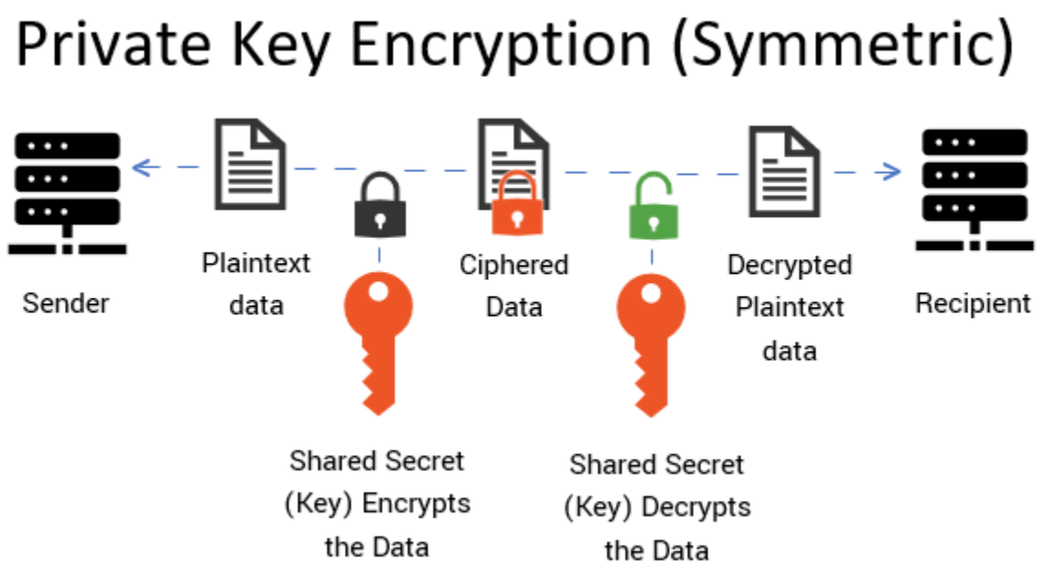

Figure 1: Private Key Cryptosytem

   The data is encrypted as well as decrypted using the shared secret key.

2. **Public Key Cryptosystem:**
   In Public Key Cryptosystems here is a pair of keys, one is private and the other is public, present with the both, receiver and the sender. The encryption process is done using the receiver's public key and the decryption is done using the receiver's private key, also called asymmetric key cryptography. Example - RSA, Knapsack, Elliptic Curve Cryptography, McEliece Cryptosystem, NTRU, Lamport Signature Scheme(refer Figure.2).
   The data is encrypted using the receiver's public key and decrypted using the receiver's private key, the pair of keys are distinct but are mathematically related.

3. **Hash Function:**
   Hash functions map data of arbitrary size to data of fixed size, and the value returned by a hash function is called hash values, hash codes, digest, or simply hashes. Hashing is a method of one-way encryption. Example – SHA-256, XMSS,LMS(Leighton Micali),BPQS(refer Figure.3).

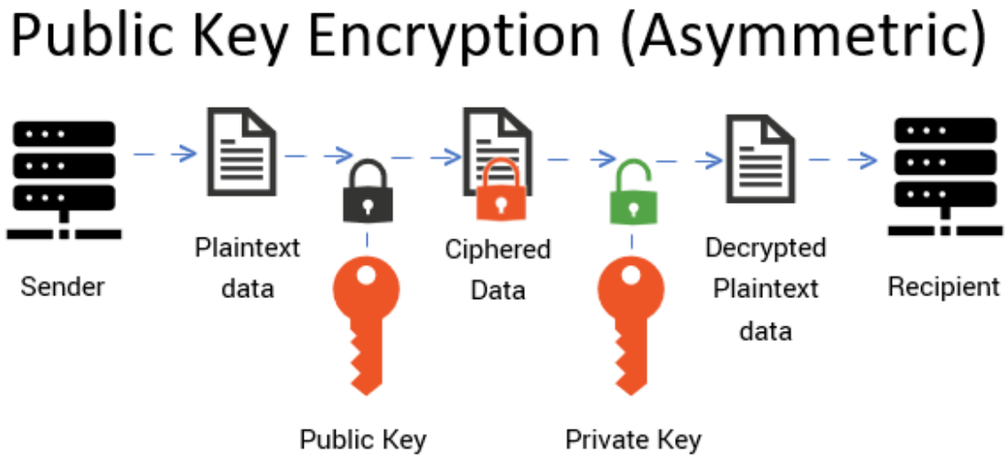

Figure 2: Public Key Cryptosytem

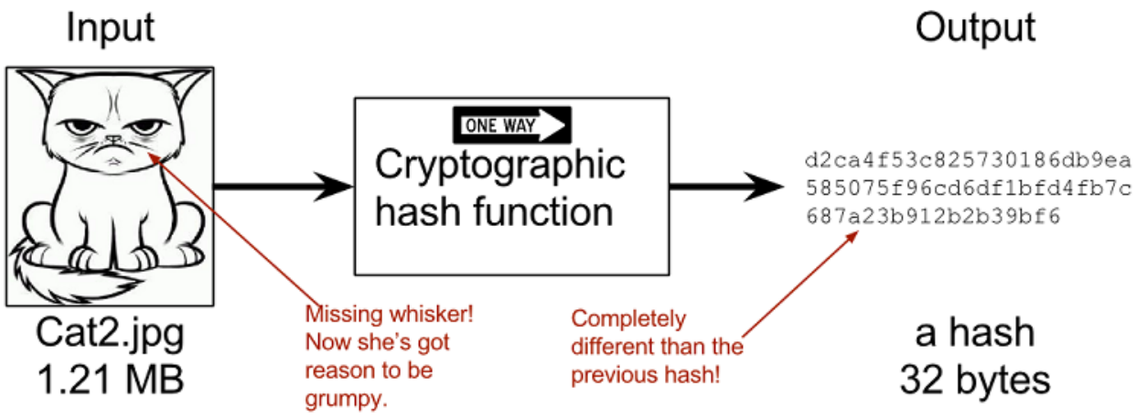

Figure 3: Cryptographic Hash-Function

The one-way cryptographic hash function maps the originals image of the cat into some random hashes.
Much of the classical cryptosystem was based on the two mathematical principles which are perfectly secure before the advent of quantum computing and the famous Shor's algorithm which had revolutionized the era of computing but had also put the global security at risk. The two main mathematical principles which from the root of classical cryptosystems are

1. **Factoring problem:**
   Given $n=p*q$ where n is made public and if p and q being two large primes factoring n into p and q is quite infeasible for the classical computer using bits. The famous RSA scheme developed by Rivest, Shamir, and Adleman at MIT, the original paper was *"A Method for Obtaining Digital Signatures and Public-Key Cryptosystems"* relies on this factoring problem.
   For example, let us consider the scenario of the RSA scheme and determine how the security issue of RSA lies completely on the factorization problem and for this let us look at the Key Generation step.
   In any public-key cryptosystem like RSA, the encryption process is done using the public key generated by the receiver, and the decryption is done by the private keys generated by the receiver and hence the key generation is to be carried out by the receiver solely. For RSA the key generation and encryption and decryption is carried out through the following mechanism
   Here Public Keys are (n,e) and private keys are (p,q,d).
   It's now obvious that if anyone can generate p and q from n, $\Phi(n)$,e and consequently d will be known and the whole security will be broken. For example, let assigned 3 and 5 to the value of p and q respectively, n will be then 15. Now n is public and the prime factorization of n revealed that p and q can only take the value of 3 and 5 and

Algorithm 1: RSA

(a) **Key Generation**

- Choose two large prime p and q and compute n as $n = p * q$
- Determine the value of Euler Phi function over n ($\Phi(n)$) to find the number of a positive integer that is less than n and are relatively prime to n as $\Phi(n) = (p-1) * (q-1)$
- Choose an integer e(encoding number) such that e and $\Phi(n)$ are relatively prime, $gcd(e, \Phi(n)) = 1$
- Determine the valued such that d(decoding number) is the multiplicative inverse of e under mod $\Phi(n)$

(b) **Encryption** Encrypt the message m using the encoding number e as $m\prime = m^e$ mod n

(c) **Decryption**

- Decry-pt the encoded message $m\prime$ using the decoding number d by first calculating $(m\prime)^d$ mod n = $(m)^{ed}$ mod n
- Applying the basic theorem of modular arithmetic relates ed to $\Phi(n)$ as $ed \equiv 1$ (mod $\Phi(n)$) $\Rightarrow$ ed mod $\Phi(n) = 1 \Rightarrow$ ed = k$\Phi(n)$ + 1,where k is the quotient.Hence $(m)^{ed}$ mod n = $(m^{\Phi(n)}$ mod n$)^k$m
- Applying Euler's Theorem $m^{\Phi(n)}$ mod n = 1,$(m\prime)^d$ mod n = 1^k.m = m(original message)

thus RSA is broken and thus we need two large primes for the value of p and q such that there is no scope to determine the value of p and q from n i.e. factorization of n doesn't reveal the value of p and q.

2. **Discrete Logarithmic Problem:**
Given an instance, a large prime p, a generator of the group (Z_p^*,*) say g where $Z_p^* = \{x : gcd(x, p) = 1$ and x $\neq 0\}$ and let A be an element $\in$ Z_p^* and finding an integer b such that $0 \leq b \leq p$–2 such that $A = g^b$ mod p is indeed hard for the classical computer using bits when the number involved is quite large. The famous Diffie Hellman Key Exchange Protocol is based on this principle of Discrete Logarithm.
For example, let us consider the scenario of Diffie Hellman Key Exchange Protocols and determine how the security issue of this scheme depends completely on the Discrete Log Problem i.e. how the Keys are to be shared between the two parties without compromising the security. Note that Diffie Hellman Key Exchange protocol is only a protocol for key sharing and it's not any complete Cryptosystem with proper encryption and decryption algorithm usable for sharing secret messages over an insecure channel. For Diffie Hellman, Key exchange is carried out through the following mechanism.

Algorithm 2: Diffie Hellman Key Exchange Protocol

(a) Any of the two parties choose a large prime say p, then $(Z_p^*, *)$ would be a cyclic group with a generator say g,(p,g) is made transferred to the other party over the public insecure channel.

(b) Each of the two parties chooses two integers to say a,b respectively which are secret to them only such that $0 < a(X_a) or b(X_b) < p - 1$ and calculate $A(Y_a) = g^a$ mod p and $B(Y_b) = g^b$ mod p and transferred A and B over the public insecure channel to the other party.

(c) Now each of the two-party has two things one is the calculated value of A or B and another one is the received value of B or A from the other party respectively.

(d) Each of the two-party computes common Secret Key $K = (B)^a$ mod p = $(A)^b$ mod p = g^{ab} mod p.

Here (p,g,A,B) are the public keys, and (a,b) are private keys. Note that in Diffie Hellman Key Exchange Protocol there is no active need of sharing the secret key but

the two-party can compute the common secret key of their own from the private(a,b) and public(p,g,A,B) keys. The actual encryption and decryption are to be carried out by the common secret key K. The adversary had access only to (p,g,A,B), and the adversary cannot decrypt the ciphertext unless and until he or she has the common secret key K from which the encryption is carried out i.e. the adversary has access to the private keys (a,b)(refer Figure.4).
The security of Diffie Hellman Key Exchange protocol lies on the Discrete Logarithmic Problem of A or B i.e. the attacker cannot compute a or b given $A = g^a$ mod p or $B = g^b$ mod p.

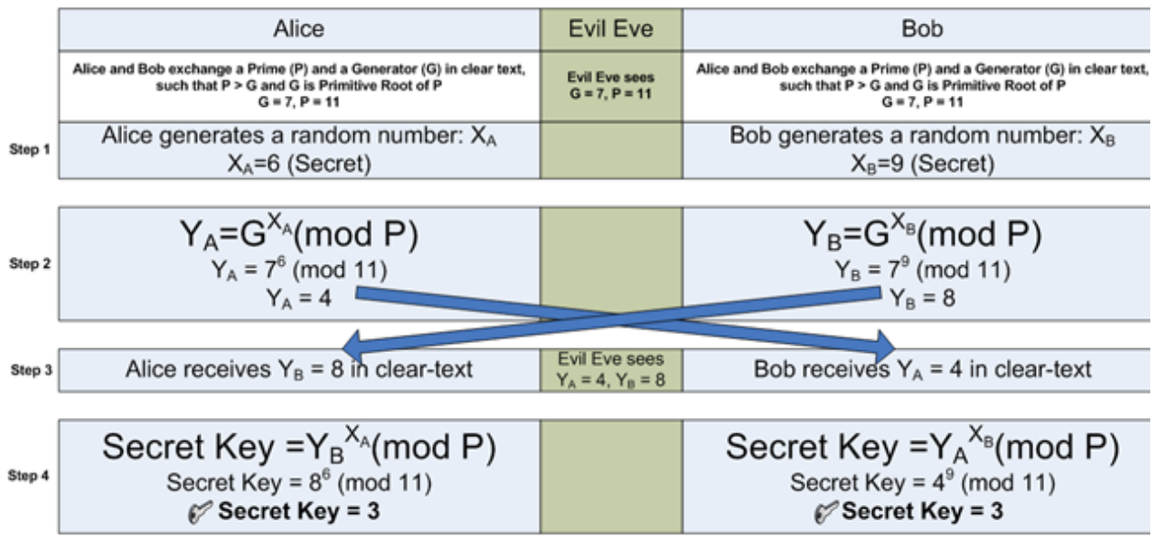

Figure 4: Diffie-Hellman Key Exchange Protocol

Here in the above example, the prime number chosen is 11 and the generator of the group will be then 7, $A(Y_A)$ is computed to 4, and $B(Y_B)$ is computed to be 8 and common key K is calculated to 3. In any classical computer, both the factorization as well as the discrete logarithmic problem are infeasible to compute.

A closer look at the Shor's algorithm will determine how Shor's algorithm implements the principle of Quantum Mechanics to find the factors of large prime. Shor's algorithm is based on the fact that the factoring problem can be reduced to finding the period of the function and its where the Quantum Computing came into the realm and actual quantum speedup over finding the period of the modular function $f_{a,N}(x)$ with the help of Q.F.T. Note that not all the step involved in the Shor's algorithm need the use of a quantum computer except for the step to finding the period of the modular function $f_{a,N}(x) = a^x$ Mod N $\forall x \in \mathbb{N}$ and $x \leq q$ need the use of Quantum Computer with its ability to be in a superposition to calculated $f_{a,N}(x)$ for all needed x for large N of about 100 digits long.
The basis of finding the period of a periodic function can be traced back to Simon's Periodicity Algorithm which is all about finding a pattern in periodic function.

2.2 Simon's Periodicity Algorithm

Simon's algorithm is a combination of both classical as well as quantum procedure.
Suppose we are given a function $f : \{0,1\}^n \to \{0,1\}^n$,that can be calculated but are treated as a black box, there exists a hidden binary string $c = c_0, c_1, c_2, \ldots, c_{n-1}$,we have

$$f(x) = f(y) : y \oplus c = x \forall x, y \in \{0,1\}^n \tag{1}$$

where $\oplus$ is the bit-wise exclusive-or operation.Thus the periodic function repeats itself in some pattern and the pattern is determined by c,c is called the period of the function, the goal of the Simon algorithm is to find the value of c(refer Table.3).
Example: Lets us work out an example with n = 3 and c = 101, we then have the following value of x,y,c.
One might think that finding the period of a periodic function to be an easy task. But that is true only for when thinking periodic function of slowly varying continuous type (for example

Table 1: Simon's Periodic Table for n = 3 and c = 101

Y	**C**	$X = Y \oplus C$
000	101	101
001	101	100
010	101	111
011	101	110
100	101	001
101	101	000
110	101	011
111	101	010

sin function) whose value at a small sample of points within a period can give powerful clues about what the period might be. But periodic function which gives a random value from one integer to the next within the period c gives no hint for determining the value of the period c. [9]

2.3 Shor's Algorithm

Peter Shor of AT&T Lab invented a quantum algorithm for efficiently finding the prime factors of a large number, classical factoring algorithm has an order of $O((\log N)^k)$ whereas Shor's algorithm has an order of $O(\log N)$.
One thing to be kept in mind that the Shor algorithm is for factoring non-prime integer, there exists a much polynomial-time algorithm for integer multiplication but no polynomial-time algorithm for factoring. Quantum algorithms are faster than a classical algorithm and are based on Quantum Superposition and Quantum Fourier transform(QFT).
The run time of factoring algorithm on any classical computer is of exponential order of $O(exp[L^{1/3}(logL)^{2/3}])$ and the same on any quantum computer is of polynomial order $O((L)^3)$, where L is the total number of bit in integer N.
Before looking into the details of how Quantum Computing can efficiently calculate the periods let us examine how knowing the period can eventually lead to the breakage of the Factorization Problem and Discrete Logarithmic problem.
Apart from evaluating the factors from the periods (as discussed in detail in upcoming sections) let us look at an alternative to decrypt the ciphertext exploiting the knowledge of the periods. Before that, it is important to know the basics of the steps involved in the encryption and decryption process.
The following two tables summarizes the RSA encryption and decryption without factorization "Table 2" summarizes information available with the communicating party and "Table 3"summarizes the breaking of RSA using Quantum Computing,the key generation step has to be carried out by the receiver, he or she chooses the encoding number e to have an inverse d modulo $\Phi(n)$, where d is called the decoding number. Since m' is a power of m and vice-versa, each has the same order in G_n, say r. Now the receiver has chosen the encoding number e in such a way that it has no factor common with $\Phi(n)$. Since the encoded message m' is in G_n, its order r is a factor of the order of (p-1)(q-1) or $\Phi(n)$ of G_n.
So e have no common factor in common with r, and therefore it has an inverse say d' modulo r, thus

$$(m')^{d'} \equiv m^{ed'} \equiv m^{1+ar} \equiv m \mod N$$

where every member v of a finite group G is characterized by its order r, the smallest integer for which(in case of G_N). [10]

$$v^r \equiv 1 \mod N$$

Shor algorithm is dependent on two mathematical principle

1. **Modular Arithmetic** The basis of an algorithm for finding the pattern or the period in a periodic function had been covered in the previous section under Simon's periodicity algorithm, in Shor's algorithm one has to calculate the period of a modular function $f_{a,N}(x) = a^x \mod N \; \forall x \in \mathbb{N} and x \leq q$, where a is co-prime to the given number N and is less than N but does not have a nontrivial factor in common with N, the testing of

Table 2: Available information with the sender and receiver

Sender	**Reciever**
•m	• m'
• e	• e and d satisfying $ed \equiv 1 \bmod \Phi(n)$
•n(n=p*q)	• p and q
• $m' \equiv m^e \bmod$ n	• $m \equiv (m')^d \bmod$ n

Table 3: Breaking RSA using Quantum Computing

Adversary without Q.C.	**Adversary with Q.C.**
• m'	• m'
• e	•e
•n(n=p*q)	•n(n=p*q)
	•Quantum computer calculate r of $(m')^r \equiv 1 \bmod$ n
	•Classical Computer computes d' such that $cd' \equiv 1 \bmod$ r
	• $(m\prime)^{d'} \equiv m^{ed'} \equiv m^{1+ar} \equiv$ m modulo N

such a factor can be done with help of Euclid algorithm to calculate the GCD(a, N) if GCD computes to 1, it's all fine a is co-prime to N and we can go ahead but if the GCD(a, N) does not compute to 1,i.e. a is not co-prime to N we got the factor of N and we are all done. There exists a theorem in classical number theory that for any co-prime $a \leq N$, the above modular function $f_{a,N}(x) = a^x \bmod$ N $\forall x \in \mathbb{N}$ and $\leq q$ will output a 1 for some r<N,where r is called the period of the function f().After hitting 1 the value of the function will simply repeats,i.e. $f_{a,N}(r+s) = f_{a,N}(s)$

2. **Quantum Fourier Transform (Q.F.T.)** The Heart of Shor's algorithm is a super-fast Quantum Fourier Transform which can be carried out by a spectacularly efficient quantum computer built entirely out of 1-Qbit and 2-Qbit gates, is a linear transformation on quantum bits or Qbits and is the Quantum analog of Discrete Fourier transform(DFT), where DFT converts a finite sequence of equally-spaced samples of a function into the same length sequence of equally-spaced samples of the discrete-time Fourier transform(a form of Fourier analysis that applies to a sequence of values, operates on discrete data, often samples whose interval has a unit of time) which is a complex-valued function of frequency. The discrete Fourier transform a sequence of N complex numbers $\{x_n\} = x_0, x_1, x_2, ..., x_{n-1}$ into another sequence of complex numbers,$\{X_n\} = y_0, y_1, y_2, ..., y_{n-1}$,which is defined by

$$y_k = \sum_{n=0}^{N-1} x_n . e^{\frac{-i2\pi}{N}kn} = \sum_{n=0}^{N-1} x_n [\cos(\frac{2\pi}{N}kn) - i\sin(\frac{2\pi}{N}kn)] \tag{2}$$

The Quantum Fourier transform acts on a quantum state$\sum_{i=0}^{N-1} x_i|i\rangle$and maps it to the quantum state $\sum_{i=0}^{N-1} y_i|i\rangle$ according to the formula

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j . e^{\frac{i2\pi}{N}kj} \tag{3}$$

Note that only the amplitude of the state was affected by this transformation.

Finally, we are now in a position to recollect all the fact and describe the Shor's algorithm with an example of an odd integer n say 15 and apply Quantum Fourier Transform

1. Determine whether the given number is an even, prime or an integral power of a prime number using an efficient classical algorithm if it's then we will not use the Shor's algorithm

2. Choose an integer q such that $n^2 < q < 2n^2$ let say 256 ($225 < 256 < 450$)

3. Choose a random integer x such that gcd(x,n)=1 let say 7(7 and 15 are co-prime)

4. Create two quantum register, one input and other is the output that must be entangled so that the collapse of the input register correspondence to the collapse of the output register

   - Input register: Must contain enough qubits to represent a number as large as q-1, here for q = 256 we need 8 qubits.
   - Output register: Must contain enough qubits to represent a number as large as N-1, here for N = 15 we need 4 qubits.

5. Load the input register with an equally weighted superposition of all integer from 0 to q-1. 0 to 255

6. Load the output register with initial states 0 Now the total state of the system at this point will be $\frac{1}{\sqrt{(256}} \sum_{a=0}^{255} |a, 000\rangle$

7. Apply the modular function $f_{a,N}(x) = a^x$ Mod N $\forall x \in \mathbb{N}$ and $x \leq q$ to each number in the input register, storing the results of each computation in the output register. The following table summarize the state of Input and Output register for n=15,q=256,x=7(refer Table.4).

Table 4: States of the input and output register after applying the modular function $f_{x,N}(a) = x^a$ Mod N $\forall \in \mathbb{N}$ and $a \leq q$ for x = 7 and n =15 and a is the input state

Input Register	$f_{7,15}(x) = 7^x$ **Mod 15** $\forall x \in N$	**Output Register**
$\|0\rangle$	$f_{7,15}(x) = 7^0$ Mod 15 $\forall x \in N$	1
$\|1\rangle$	$f_{7,15}(x) = 7^1$ Mod 15 $\forall x \in N$	7
$\|2\rangle$	$f_{7,15}(x) = 7^2$ Mod 15 $\forall x \in N$	4
$\|3\rangle$	$f_{7,15}(x) = 7^3$ Mod 15 $\forall x \in N$	13
$\|4\rangle$	$f_{7,15}(x) = 7^4$ Mod 15 $\forall x \in N$	1
$\|5\rangle$	$f_{7,15}(x) = 7^5$ Mod 15 $\forall x \in N$	7
$\|6\rangle$	$f_{7,15}(x) = 7^6$ Mod 15 $\forall x \in N$	4
$\|7\rangle$	$f_{7,15}(x) = 7^7$ Mod 15 $\forall x \in N$	13

•
•
•

8. Now the previous approach of applying the modular function and storing the content of register in a tabular form is good but it's not efficient when N is large, we are now in a position to implement the principle of Quantum Mechanics to calculate the period. Measure the output register, this will collapse the superposition to represent just one of the transformation after the modular function, let call it c.
   Our output register will collapse to represent just one of the following $|1\rangle$,$|4\rangle$,$|7\rangle$, or $|13\rangle$.For the sake of simplicity let assume that its $|1\rangle$.

9. Since the two register are entangled, measuring the output register will have a direct impact on the state of the input register and would lead it to partially collapse into an equal superposition of each state in between 0 and q-1 that yield c,where c is the value of the collapsed output register.
   In our example,our output register collapse to $|1, \rangle$,the input register will then collapse to $\frac{1}{\sqrt{(64}}|0\rangle + \frac{1}{\sqrt{(64}}|4\rangle + \frac{1}{\sqrt{(64}}|8\rangle + \frac{1}{\sqrt{(64}}|12\rangle, \ldots$ The probabilities in this case are $\frac{1}{\sqrt{(64}}$ since our register is now in an equal superposition state of 64 value (0,4,8,12,...,252). Let now apply Quantum Fourier Transform(QFT) on the partially collapsed input register.The QFT will take a state as input say it be $|a\rangle$ and transform it into

$$\frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} |c\rangle . e^{\frac{i2\pi}{q} ac}$$

   The final state after applying the QFT will be $\frac{1}{\sqrt{m}} \sum_{a \in A} |a\rangle, |c\rangle$, where m is the

cardiniality of the set A of all the value that the modular function yields to the value of collapsed output register and $|a\rangle$ corresponds to

$$\frac{1}{\sqrt{q}}\sum_{c=0}^{q-1}|c\rangle.e^{\frac{i2\pi}{q}ac}$$

In our example q = 256, c=1, A=0,4,8,12,...,252, m=n(A)=64,so the final state of the input register after QFT

$$\frac{1}{\sqrt{64}}\sum_{a\in A}\frac{1}{\sqrt{256}}\sum_{w=0}^{255}|w\rangle.e^{\frac{i2\pi}{256}aw},|1\rangle$$

The QFT will essentially peak the probability amplitudes at an integer multiple of q/r where r is the desired period in our case r is 4. $|0\rangle, |64\rangle, |128\rangle, |192\rangle, \ldots$ So we are no longer have an equal superposition of states. Measure the state of register one, call this value t and this value will have a very probability of being a multiple of q/r. With our knowledge of q and t, there are several methods of calculating the period(one method is the continued fraction expansion of the ratio between q and m.)

10. Now that we have the period, the factor of N can be determined by taking the greatest common divisor of N with respect $x^{\frac{r}{2}}+1$ and $x^{\frac{r}{2}}-1$. The idea here is that this computation will be done on a classical computer. In our example r is 4,thus the $gcd(7^{\frac{4}{2}}+1,15)=5$ and $gcd(7^{\frac{4}{2}}-1,15)=3$ and thus the factor of 15 are 5 and 3.

Algorithm 3: Shor Algorithm

Result: Factors of n
Choose an integer less x ;
if $X \leq n$ *and gcd(x,n) = 1* **then**
  Use the QFT to determine the unknown period r of the modular function $f_{x,n}(a) = x^a \bmod n \ \forall a \in N$;
  if *r=even* **then**
    Use Euclid's algorithm to calculate $GCD((x^{r/2}+1), N)$ and $GCD((x^{r/2}-1), N)$;
  else
    Choose new x;
  end
else
  Choose new x;
end

2.4 The fallacy of the classical cryptosystems and the advent of a quantum computer

After discussing Shor and Simon algorithm in detail we are now in a position to explain how the quantum computer employs the principles of quantum mechanics in computation and how all these principles supports the quantum Fourier transform that form the heart of the Shor algorithm and make it possible to factorize large number which would otherwise infeasible for a classical computer do not employ the principle of quantum mechanics. Before that its important to discuss three things

1. **Quantum-bits(Qubits)**: Classical bits or smallest possible block of information in any classical computer can exist in only two binary states 0 or 1 i.e. either true or false. A physical implementation of bits would be using two energy levels of the atom. An excited atom with its electron in higher energy level than in the ground state denotes the 1 state and the atom in the ground state with all its electrons in their grounds state denotes 0 states. These states are denoted by the Dirac ket notation, state 0 is denoted by $|0\rangle$, and state 1 is denoted by $|1\rangle$. Quantum Mechanics allows the superposition of $|0\rangle$ and $|1\rangle$ each with its amplitudes. This is what is called Qubits, i.e. Qubits are a superposition of $|0\rangle$ and by $|1\rangle$.

- 1 qubits or superposition of 2 possible state $|0\rangle$ and $|1\rangle$
- 2 qubits or superposition of 4 possible state $|00\rangle$, $|01\rangle$, $|10\rangle$ and $|11\rangle$
- 3 qubits or superposition of 8 possible state $|000\rangle$, $|001\rangle$, $|010\rangle$, $|011\rangle$,...
-
-
-

n qubits or superposition of 2^n possible state Describe by a "Wavefunction" i.e. vector of 2^n amplitudes.

2. **Quantum Superposition:** Quantum superposition is a fundamental principle of quantum mechanics, it states that any two quantum states can be added or superimposed and the results would be another valid quantum state. A pure qubits state is a coherent superposition of the basic state. This means that a single qubit can be described by a linear combination of $|0\rangle$ and $|1\rangle$

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{4}$$

where α and β are probability amplitude and in general both are the complex number.According to born rule, probability of $|0\rangle$ with value 0 is $|\alpha|^2$ and the probability with outcome $|1\rangle$ with value 1 is $|\beta|^2$.Because the absolute square of the amplitude equate to probability it must be true that

$$|\alpha|^2 + |\beta|^2 = 1$$

Note that the qubit superposition state does not have a value in between 0 and 1 rather there is a probability of $|\alpha|^2$ that it attains 0 state and a probability of $|\beta|^2$.In other words, superposition means that there is no way, even in principle, to tell which of the two possible states forming the superposition state pertains.

3. **Quantum Entanglement:** Quantum entanglement is a physical process that occurs when a pair or group of particles is generated or interact in such a way that the quantum state of each particle of the pair cannot describe independently of the state of other in the pair. The simplest system to display quantum entanglement is the system of two qubits, two entangled qubits

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

in this state called equal superposition there are equal possibility of measuring either the product state with $|00\rangle$ or $|11\rangle$ as $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$,i.e.there is no way to tell that whether the first qubit has the value of 0 or 1 and same with the case second bit.

Now the things are going interesting till now we have discussed much mathematical stuff, we deal with QFT, Simon algorithm, Quantum entanglement, quantum superposition and now itis time to wind up everything and formally state why classical cryptosystem which fundamentally relies on Factoring Problem and discrete log problem is no longer safe in the post-quantum era.
In this paper, it had been primarily focused on the factorization problem using the Shor period finding algorithm to break the heavily used and well renowned RSA. Shor algorithm can also be modified to solve the discrete logarithmic problem using hidden subgroup problems and group homomorphism.
RSA relied on the factorization problem of finding factors of product of two large prime (here p and q).
The Shor algorithm in its final step calculate the factor by calculating $gcd(x^{\frac{r}{2}}+1, n) = a$ and $gcd(x^{\frac{r}{2}}-1, n) = b$,where a and b are the factor and r is the period of the modular function $f_{x,n}(a) = x^a$ Mod n $\forall a \in \mathbb{N} : 0 \leq a < q$ where x is the random integer $\leq n$.
Now for calculating the period r instead of proceeding classically through exhaustive search method which is not feasible for practically large n when the chosen p and q are large,we proceed with the help of Quantum Fourier Transform (QFT) to calculate the period.
QFT will be applied on the partially collapsed input registers to convert it final state to

$$\frac{1}{\sqrt{m}} \sum_{a \in A} \frac{1}{\sqrt{q}} \sum_{w=0}^{255} |w\rangle . e^{\frac{i2\pi}{q} aw}, |c\rangle$$

where
m = cardinality of the set A of all the value that the modular function yields to the value of collapsed output register
q = $n^2 \leq q \leq 2n^2$
c = value of the collapsed output register
QFT will peak the probability amplitudes at an integer multiple of q/r where r is the desired period, which we want. Measurement of the state of the registered one says it comes to be t, by continuous fraction expansion of the ratio of q and t one can find though this can be done classically and the last step of finding the period seems not that much dependent on Quantum computer.
The only quantum dependent step is peaking the probability amplitudes at an integer multiple of q/r where r is the desired period of the collapsed input register which happened only due to the presence of quantum entanglement between the input and output register which is not possible in case of a classical computer implementing bits and the collapsing of the output register to just one of the superimposed state after the application of the modular function and the input register would not be possible to load with the equally weighted superposition of all value between 0 and q-1 without qubits and the principle of quantum superposition as the classical bits does not exist as the superimposed value of 0 and 1(refer Figure.5).

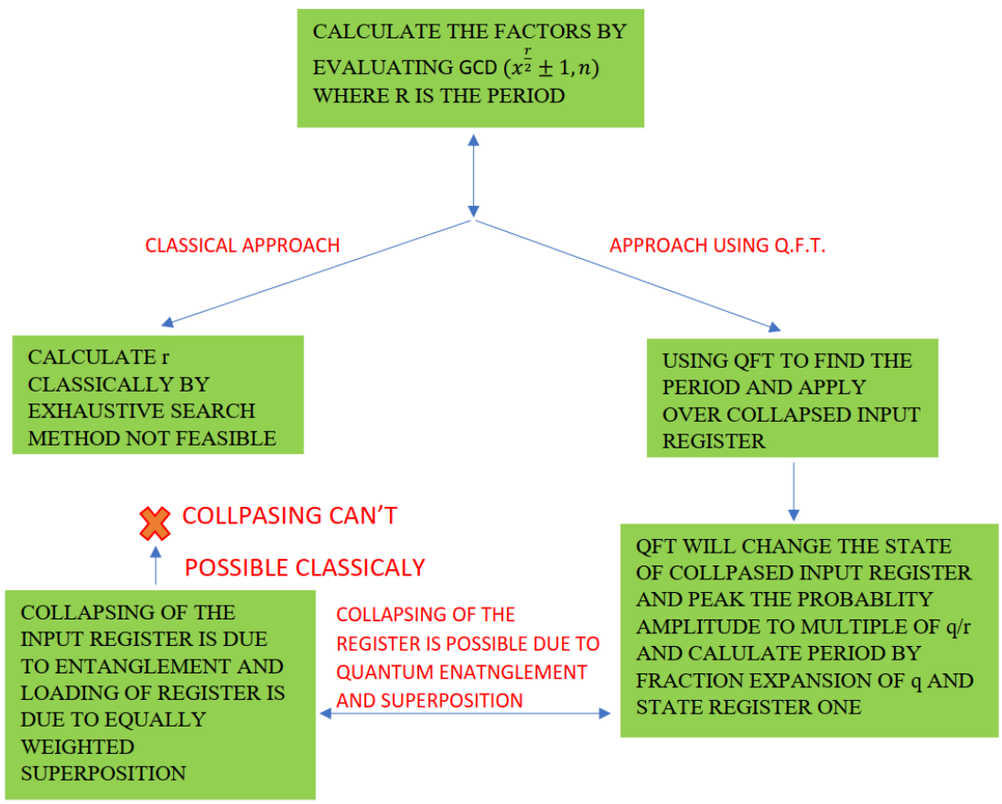

Figure 5: Fallacy of the Classical Cryptosystem

2.5 Post Quantum Cryptosystems

1. **Hash-based Signature Scheme:** Hashed Based Cryptographic approaches that rely on the hash function for security purposes are so far limited to the digital signatures schemes. Digital Signature Scheme provides a means for the authentication of the sender rather than providing security to the message. The entire security relies on the underlying hash functions.

2. **Lattice-based Cryptosystem:** Lattice-based Cryptosystem is the generic term used for the construction of cryptographic primitives which involves lattice either in the construction of the cryptosystems itself or in providing security to the cryptosystems. [11] Mathematically lattice is a subset of $\mathbb{R}^n$ or the set of all the integer linear combinations

of the basics vectors $b_1, b_2, b_3, ..., b_n \in \mathbb{R}^n$ thus

$$L = \{\sum a_i b_i : a_i \in \mathbb{Z}\} \tag{5}$$

where b_i are linearly independent vector over $\mathbb{R}$ and the ordered n-tuples $(b_1, b_2, b_3, ..., b_n)$ is the basis of the lattice. The entire security of the Lattice-based Cryptosystem lies on the hardness of some problems.
Some general Lattice problems used in the Cryptographic primitives are

(a) **Shortest-Vector Problem**

(b) **Closest-Vector Problem**

(c) **Shortest Independent Vector Problem**

3. **Multivariate Cryptosystem:** Multivariate Cryptosystem is based on the multivariate polynomials i.e. polynomials with more than one indeterminant or variable, usually quadratic over a finite field $\mathbb{F} = \mathbb{F}_q$ with q elements. [13] Let us have a system of m multivariate quadratic polynomial with n variables.

$$\begin{aligned}
p^{(1)}(x_1, x_2, ..., x_n) &= \sum_{i=1}^{n} p_{ij}^{(1)} x_i \cdot x_j + \sum_{i=1}^{n} p_i^{(1)} \cdot x_1 + p_0^{(1)}, \\
p^{(2)}(x_1, x_2, ..., x_n) &= \sum_{i=1}^{n} p_{ij}^{(2)} x_i \cdot x_j + \sum_{i=1}^{n} p_i^{(2)} \cdot x_1 + p_0^{(2)}, \\
&\vdots \\
p^{(m)}(x_1, x_2, ..., x_n) &= \sum_{i=1}^{n} p_{ij}^{(m)} x_i \cdot x_j + \sum_{i=1}^{n} p_i^{(m)} \cdot x_1 + p_0^{(m)}
\end{aligned} \tag{6}$$

The entire security of the Multivariate Cryptosystem depends on the hardness of a problem, MQ(abbreviated from Multivariate Quadratic) which can be simply stated in the following form.
Given a system of m multivariate quadratic polynomial $p^{(1)}(x), ..., p^{(m)}(x)$ in n variables $x_1, ..., x_n$ the problem is to find a solution to the system of polynomial i.e. find a vector $\overline{x} = (\overline{x}_1, ..., \overline{x}_n)$ such that $p^{(1)}(\overline{x}) = ... = p^{(m)}(\overline{x}) = 0$ The solution of the above systems of a polynomial (MQ Problem) is proven to be non-deterministic polynomial time(NP) hard over any field and is believed to be hard on average for both Classical and Quantum Computers.

4. **Code-based Cryptosystem:** Code-base Cryptography is the fourth candidate along with Hashed based Digital Signature Scheme, Lattice-based cryptosystems, and Multivariate Cryptosystems that are proven to be safe from Quantum attacks. [12] Formally Code-based Cryptosystems can be defined as those family of Cryptosystems both symmetric as well as asymmetric whose security lies entirely upon the hardness of decoding of an encoded message in linear error-correcting codes, chosen with a particular structure, for instance, Goppa code which is used in the well-known McEliece cryptosystem which is the first proposed code-based cryptosystem. The original idea behind McEliece Cryptosystem was to use an encoded-word of some chosen linear error-correcting code(binary Goppa code) to which random errors are added to obtain the ciphertext(refer Figure.6).
Error-correcting codes are that set of codes which can be used to detect as well as rectifying the error, which may have occurred either due to unavoidable physical alteration happened during the transmission of the message in the channel or by intention by some adversary. The correction of the error is done by adding some extra information called redundancy that makes it easier to detect and rectify the error. [12] In coding theory, the redundancy is also called as parity or parity check.
An arbitrary basis of the code(generator matrix) is used as the public key for the encryption purpose to encode to which further error would be added to get the final ciphertext. [2] Authorized persons can use a fast trapdoor function to remove the error and decode the intermediate codeword to obtain the original message.
The entire security of code-based cryptosystems is based on two computational aspects

(a) **Generic Decoding problem :** Decoding problem is hard on average even for a quantum computer.

(b) **Identification of the generator matrix :** The public key generator matrix is hard to distinguish from a random matrix.

Even McEliece is safe in the pre-quantum era yet it is not preferred over RSA due to its large key sizes.

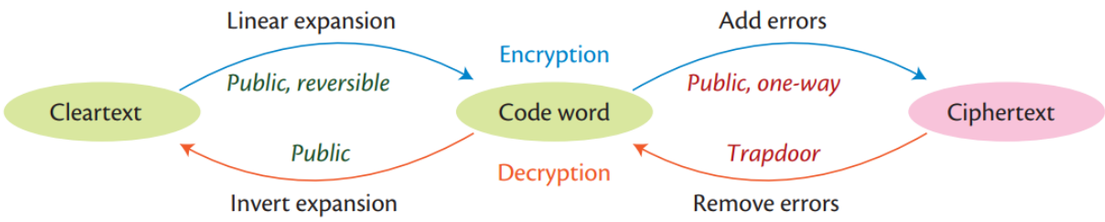

Figure 6: McEliece Cryptosystem

2.6 Signature Schemes

In most of our day to day activity apart from the privacy of our messages or the information that we share we also need the authentication of the sender where the privacy of the message is not a prime factor, we don't focus on the privacy of the messages or information being shared but focus on what we received and accept must be from that person we wanted and not form any anonymous one. In the physical world, we use the signature scheme where the sender's signature is a part of the message, which provides the authentication of the author.The message not necessary be private and the authentication of the sender can be verified by the receiver by comparing the signature contained in the shared document with some pre-existing copy of the sender signature.It can be the ID card already there with the receiver and this prevents the adversary to send any false information.
In the electronic sense, we achieved the same through signing the document with some secret key that is only available with the sender and a public key for anyone to verify the signature with some additional information say time and place so that if anyone copies the signature it is no longer a valid one.
The general algorithm of how the signature scheme work is as follows:

1. **Key Generation:**
   The sender develops a pair of a key using some Trapdoor One Way Function one is being kept secret and the other is made public.

2. **Signature Generation:**
   The sender signs the document using the secret or private key to develop the signature of message $m : s = d_{sk}(m)$, using the decryption function.

3. **Signature Verification:**
   The receiver verifies the signature of the sender by using the public verification key of the sender to develop a temporary message of the signature $s : z = e_{pk}(s)$ and then verifying with the available message m.

Note that we are no longer interested in the privacy of the message but are more interested in the authentication of the sender.

2.7 Attacks on Signature Scheme

Attack on the Signature Scheme can be broadly classified into two main categories [2]

1. **Key Only Attack:** The adversary knows only the sender's public key

2. **Message Attack:** The adversary knows some of the messages or signature apart from the sender's public key. Message attack can be classified into the following categories

- *Known message attack:* Adversary knows the signature of a message set $m_1, m_2, ..., m_n$ that is NOT chosen by the adversary.
- *Generic chosen message attack:* Adversary can get the signature of a message set $m_1, m_2, ..., m_n$ been chosen.The messages sent do not depend on the sender's public key and are chosen before the adversary had got any signature.
- *Directed chosen message attack:* Adversary can obtain the signature of a message set chosen by him/her and have been chosen before the adversary had seen any signature but, here these may depend on the sender's public key.
- *Adaptive chosen message attack:* The adversary can request the sender r for the signature that depends on the sender's public key and additionally it also depends on previously obtained signatures.

There is one more attack that's is of very little value but cannot be ignored as the attacker may use it and it depends on the development of a fraud pair of signature s and message m and can use it over the channel to mislead the receiver. The attacker may arbitrarily choose a value of the signature let $s\prime$ and use the sender's public key to develop a false message $m\prime$ that hadn't been originally sent by the sender and can mislead the receiver. $m\prime = e_{pk}(s\prime) and (m\prime, s\prime)$ is a valid pair.

Symptoms of the broken signature scheme

1. **Total break:** an adversary can recover the sender's private key
2. **Universal Forgery:** The adversary had developed or find an efficient signing algorithm functionally equivalent to the sender's signature algorithm which uses the secret key to sign the documents with help of equivalent, possible different trapdoor information.
3. **Selective Forgery:** The adversary can sign any message of his or her choice.
4. **Existential Forgery:** The adversary can create a valid signature for at least one message chosen by another person.

The invention of Grover's algorithm had proved that these classical cryptosystems for developing signature are no longer secure but hash-based cryptography can be securely be used in the quantum era, and these hash-based cryptosystems when being involved in the signature scheme it provides a secure means even in the quantum era.

2.8 Hash Function

Hash function in cryptography map strings of arbitrary length to a string of fixed length of length say n which typically lies between 128 and 512 bits. For example, SHA-256 (Secure Hash Algorithm – 256) is considered to be one of the most secure Hash Functions and it is believed to be safe from the attack from the most powerful supercomputer and even quantum attack. [2] A hash function is typically denoted by $h : \{0,1\}^* \rightarrow \{0,1\}^n$

Before hash function can be used in cryptography it has to satisfy certain properties

1. **Preimage:** Given y = h(x) find a string $x\prime$ such that $h(x\prime) = y$
2. **Second preimage:** Given x, find a string $x\prime \neq x$ such that $h(x) = h(x\prime)$
3. **Collision:** Find two strings x and $x\prime$ such that $x \neq x\prime$ and h(x) = $h(x\prime)$

All of the three properties of the hash function can be broken by a brute force attack and it had been found that only 2^n application of h () can break the preimage and second preimage problems and $2^{n/2}$ application of h () is needed to break the collision problem.

The hashed function can be used in a very efficient way in the signature scheme for the author authentication all we have to do is that compute $m\prime$ = h(m) where m is the original message and $m\prime$ is computed by the use of the hash function h () and all the comparison are done on $m\prime$. The use of the hash function not only prevents the attack where the adversary creates a false pair by burdening him to solve the preimage problem but also avoids the case where the original message is very large.

Hash functions are believed to be quantum-safe but many a time they are believed to be safe from attack by supercomputer also; for example, the SHA 256 and thus the security issue of the Hashed Based Signature Scheme in the Quantum era lies with the security of the underlying hash functions.

2.9 One Time Signature Scheme

A one-time signature scheme is a set of algorithms where the key is allowed to used only once by revealing some part of the secret signature key. The basic idea is that we use a one-way function say $f : \{0,1\}^n \to \{0,1\}^n$ and the signature key x that's would be made public of n bits and compute the verification key y from the one-way function f() and do all the computations to developed the signature s using the signature key x and the hash function h() and made public the message m and the signature s and the verification is done based on computing the one-way function f () on the revealed parts of the signature key x and comparing it with the verification key y. [1]
One more factor that must be considered is that the underlying hash function h() must be collision-resistant as there may be a situation arise when the author is not honest and can choose two different messages m and $m\prime$ such that the $h(m) = h(m\prime)$ and thus two messages m and $m\prime$ have the same signature and sign m and later claim that he or she had sign $m\prime$ instead of m. The Collison resistant hash function is an utmost factor for considering Hash-Based Signature Schemes.

2.9.1 Lamport One-way signature scheme

The idea of using a hash function in a digital signature scheme originated from Lamport who propose the first practical implementation of the hash-based Signature scheme in personal communication with Diffie.

Algorithm 4: Lamport one-Way signature Scheme

1. **Key Generation**

   - Choose $2q$ random n bit string $x_{i,j}$ for $0 \leq i \leq q$ and $j \in \{0,1\}$
   - Compute $y_{i,j} = f(x_{i,j})$ for $0 \leq i \leq q$ and $j \in \{0,1\}$
   - Authenticate and make public the $y_{i,j}$ for $0 \leq i \leq q$ and $j \in \{0,1\}$

   The Secret key is $(x_{0,0}, x_{0,1}, \ldots, x_{q-1,1})$ and public verification key is $(y_{0,0}, y_{0,1}, \ldots, y_{q-1,1})$

2. **Signature Generation** Sign the message m by revealing x_{i,m_i} for $0 \leq i \leq q$

3. **Signature Verification** The signature is verified by computing $z_i = \mathrm{f}(x_{i,m_i})$ and checking that $z_i = y_{i,m_i}$ for $0 \leq i \leq q$

Here x_{i,m_i} is the revealed part of the signature.

Each of the keys can be used for a sign at most one message if two messages being signed then their exit a possibility that the attacker may develop a valid signature for a message using the information available from the two signatures For example, let m_1 and m_2 be two message as 110 and 101 respectively then using the same private signature key the signature for m1 is $(x_{0,1}, x_{1,1}, x_{2,0})$ and for m_2 is $(x_{0,1}, x_{1,0}, x_{2,1})$ and thus valid signature for any random message say $m_n = 111$, the signature is $(x_{0,1}, x_{1,1}, x_{2,1})$ which is a valid one and hence put the security at risk.

2.9.2 Merkle's development over Lamport's one-time Signature Scheme

American computer scientist Ralph Merkle had made some improvements over Lamport's one-time signature scheme. The core idea is to use a signature key of form $\mathrm{X} = (x_0, x_1, \ldots, x_{q-1})$ where x_i is random n bits instead of the typical $(x_{0,0}, x_{0,1}, \ldots, x_{q-1,1})$ as in the original scheme developed by Lamport. The signature of the message m using this new Merkle Improved Scheme is done by revealing all the x_i bits for which the $m_i = 1$ for all $0 \leq i \leq q$ [1]. In this

scheme, the number of 0 bits in the message m is added to the end of the original message m to avoid the attack where the attacker may try to develop a valid signature for the messages which doesn't have one bit in a position where m doesn't have 1 bit and define

$$k\prime = k + \lfloor \log_2 q \rfloor + 1 \tag{7}$$

for the development of the keys instead of using k as in the original Lamport Scheme. [3]

Algorithm 5: Merkle's development over Lamport's one-time Signature Scheme

---

1. **Key Generation**

   - Choose $q\prime$ random n bit string x_i for $0 \leq i < q\prime$
   - Compute $y_i = f(x_i)$ for $0 \leq i < q\prime$
   - Authenticate and make public the y_i for $0 \leq i < q\prime$

   The Secret key is $(x_0, x_1, \ldots, x_{q\prime-1})$ and public verification key is $(x_0, x_1, \ldots, x_{q\prime-1})$

2. **Signature Generation**

   - Count the number of 0 bits in message m and let called this number to be a and define ab as the binary representation of a with $q\prime$ bits and define $m\prime = (m \| ab)$ here $\|$ stands for concatenating
   - Sign m by revealing x_i for all i, $0 \leq i < q\prime$ such that $m\prime_i = 1$

3. **Signature Verification**

   - Find a and ab from m and generate $m\prime$
   - Compute $z_i = f(x_i)$ and verified the result with the public key y_i for all i such that $m_i = 1$

Here x_i is the revealed part of the secret key and now the message m had been concatenated with the number of 0 bits in m. If m has more 0 bits then the binary representation of a(the number of 0 bits in the original m)will have more 1 bits instead of 0 bits and thus attacker would not be able to sign any false messages m_o where the position of 1 bit in the m_o is same as in the original message m.

Example: Let $k = 8$ and thus $k\prime = k + \lfloor \log_2 8 \rfloor + 1 =$ k + 4 =12
and $m = (10111100)_2$ thus we have $a = 3$ and $a_b = (0011)_2$ and
$m\prime = m \| a_b = (101111000011)_2$
Thus the signature will be $s = (x_0, x_2, x_3, x_4, x_5, x_{10}, x_{11})$
Now the attacker can't develop any false signature and false message,the original message m is lost and all the algorithm is performed on $m\prime$.
The key length is now roughly $n(k + \log_2(k))$ bit long and the signature which depends on the message m but it's about $n(k + \log_2(k))/2$ on average.

It's now obvious from the above two algorithms that if the secret key is used many a time then more and more of it will get revealed and thus the probability, he that the attacker would be able to generate the public key from the exposed part of the key would increase.

There had been several proposals over the Hash-based one-time signature scheme like Bleichenbacher's and Maurer's Digital Signature based on Directed Acyclic Graph as described in their paper, "Directed acyclic Graph, One Way function and Digital Signature"; Dods, Smart, and Maurer had analyzed the proposal made by Bleichenbacher and Maurer but found that it does not work as fine as Winternitz, in their paper "Cryptography and Coding" on December 19-21 of 2005 in the 10th IMA International Conference. Bos and Chaum in the year 1992 had proposed another variant of Lamport scheme and in 2002 Reyzin and Reyzin had proposed another similar variant the key difference between Bos-Chaum and Reyzin-Reyzin was that the prior had focused upon the minimization of the public key while the later had focused upon the minimization of the signature size.

2.9.3 Winternitz One-time Signature Scheme for smaller key size

The scheme proposed by Lamport on Hash-based One Time Signature was quite efficient and easy to adopt but the main problem lies with the large key size, the key size is 2kn whereas the proposal made by Winternitz was only about n. The basic idea is to use one string in the one-time signature scheme to sign several bits in the message digest (output of the hash function) simultaneously. [3] It first appears in the thesis of Ralph Merkle, *"Secrecy, Authentication, and Public key System"*, he described that the method was suggested to him by Winternitz in the year 1979.
Like Lamport scheme Winternitz one-time signature scheme also use similar kind of one-way function

$$f : \{0,1\}^n \to \{0,1\}^n$$

and a cryptographic hash function

$$h : \{0,1\}^* \to \{0,1\}^n$$

The main difference starts with the key's generation and signature development. The following table summarises the various aspects of the one-time signature scheme that had been discussed (refer Table.5).
The one-time signature scheme introduced was not efficient for the practical scenario. In practical life, we need something of the sort many time signatures schemes that can be implemented in signing in practical life provided that the security issue remains stable and the scheme provides good memory management.
Several messages definitely can be signed with the one-time signature scheme but in that case, we have stored a very large number of verification keys which could cost a huge amount of memory, an idea that can arise is to send both the verification key and the signature key at the same time this would solve the problem of memory management but would cost to the authentication problem of the verification keys.
In the year 1979 Merkle had proposed a method that would neither cost the security nor cost memory and authentication problem. The basic idea is to use a complete binary hash tree that would reduce the number arbitrary but fixed number of one-time verification keys to the validity of one single public verification key that's the root of the hash tree.
The method describe just above was Merkle Static Tree, the number of the valid signature that can be done with it are finite say it is 2^d(for simplicity and the ease of calculation it had been taken as a power of two and also a binary tree of height h has 2^h leaves) but there is also a provision by which it allows to sign an infinite number of messages by the use of a dynamically expanding tree, which grows along with the number of signatures made. There is a problem associated with it, the size of the signature grows after each signature and these not only lead to inefficiency but it also led to insecurity as it revealed the total number of the signature being made.

2.10 Merkle Static Tree

Let the total number of the signatures made is finite, say it is 2^d,we choose a one-way hash function say $h : 0, 1^* \to 0, 1^n$. For signing 2^d messages say $(m_1, m_2, m_3, m_4, \ldots, m_{2^d})$ we need 2^d number of key pairs(s_i, p_i) for $0 \leq i < 2^d$ here d is called the height or depth of the tree. [8]

Here the tree is fundamentally built upon the one-time hash function h() if the underlying hash function is secure then the public verification key is also secure, hence the security issue hadn't been compromised and ultimately depends on the hash function and thus safe even in the quantum era. We assumed that message is signed using the one-time key pairs (s_i, p_i) for $0 \leq i < 2^d$ the question may arise how can the verification be done using the public key? For this one must authenticate the verification keys p_i using the public verification key Y, the root of the Hash Tree. This is typically done using an authentication path that follows the path from Y to the p_i.

There are two possible authentication paths for the verification of the key

1. The first part follows from the root to the leaves

2. The second part follows from the leaves to the root

Algorithm 6: Winternitz One-time Signature Scheme for smaller key size

1. **Key Generation:**

   - A Winternitz parameter $w \geq 2$ is chosen which denotes the number of bits to be signed simultaneously, then t_1 and t_2 will be computed as $t_1 = \lceil \frac{n}{w} \rceil$ and $t_2 = \lceil \frac{\lfloor log_2 t_1 \rfloor + 1 + w}{w} \rceil$ and $t = t_1 + t_2$
   - Choose t random n bit string x_i for $0 \leq i < t$
   - Compute and authenticate and make public

     $$y_i = f^{2^w - 1}(x_i)$$

     for $0 \leq i < t$
   - The Secret signature key is $(x_0, x_1, \ldots, x_{t-1})$ and public verification key is $(y_0, y_1, ..., y_{t-1})$

2. **Signature Generation:**

   - Zeroes are added to the beginning of the message m such that the total length of m is divisible by w. The extended m is spilled into t_1 w-bits strings $b_{t-1}, b_{t-2}, b_{t-3}, \ldots, b_{t-t_1}$ such that $m = b_{t-1} \| ... \| b_{t-t_1}$
   - Identify each b_i with an integer $0, 1, 2, 3, 4, 2_w$–1 and find the check sum as

     $$\sum_{i=t-t_1}^{t-1} (2^w - b_i) \tag{8}$$

     It can be shown that the length of $c \leq t_1 2^w$ and thus the length of a binary representation of c is less than

     $$\lfloor log_2 t_1 2^w \rfloor + 1 = \lfloor log_2 t_1 \rfloor + 1 + w$$

     Minimum number of zeroes are added at the beginning of c such that the length of c is divisible by w and the extended string is spilled into t_2 w-bits strings $(b_{t_2-1}, b_{t_2-2}, \ldots, b_{t_2-t_2})$ such that $(b_{t_2-1} \| b_{t_2-2} \| \ldots \| b_0)$
   - The signature is computed as

     $$s = (f^{b_{t-1}}(x_{t-1}), \ldots, f^{b_1}(x_1), f^{b_0}(x_0))$$

3. **Signature Verification:** The signature $s = (s_0, s_1, \ldots, s_{t-1})$ and computing the bit string $b_0, b_1, \ldots, b_{t-1}$ as above and checking if

   $$s = (f^{2^w - 1 - b_0}(s_0), \ldots, f^{b 2^w - 1 - b_{t-1}}(s_{t-1}))$$

Table 5: One time signature scheme

NAME OF THE SCHEME	PUBLIC KEY SIZE	SIGNATURE SIZE
Lamport Scheme	2kn	kn
Merkle's Improvement of Lamport	$n(k + \log_2(k))$	$n(k + \log_2(k))/2$
Winternitz	n	tn

Algorithm 7: Merkle's Tree for generating public key

1. **For Signing Message** For signing 2^d messages developed that many numbers of secret-public key pair (s_i, p_i) for $0 \leq i < 2^d$
2. **Building Binary Tree**
   - The leaves of the tree are computed as $y_i = h(p_i)$ for $0 \leq i < 2^d$
   - The nodes at the height j are computed as $y_j^i = h(y_{j-1}^{2i}||y_{j-1}^{2i+1})$ for $0 \leq i < 2^{d-j}$ and for $0 \leq j \leq d$
3. **Making the public key**Authenticate and make public the root of the tree $Y = y_d^0$ The root of the tree $Y = y_d^0$ is the only public key and all the key pair are secret keys (s_i, p_i) for $0 \leq i < 2^d$

As an example, let us take the authentication of p_4 as depicted in the following Merkle Tree of height $d = 3$ and thus having 8 key pairs
The first possibility follows a path from the root of the tree, Y_3^0 is an authenticate public key now,

1. The signer revealed y_2^0 and y_2^1, the verifier computes y_3^0 and if it matches with the given Y, y_2^1 is verified.
2. The signer revealed y_1^2 and y_1^3, the verifier computes y_2^1 and if it matches with the pre-verified value of y_2^1, y_1^2 is verified.
3. The signer revealed y_0^5 and the verifier computes y_0^4 from p_4 using the one-way hash function, then the verifier computes y_1^2 and if it matches with the pre-verified value of y_1^2, p_4 is verified. One can find that half of the transmission is redundant and element that are redundant i.e. y_2^1 and y_1^2

The second possibility follows a path that leaves to the root,

1. The verifier computes y_0^4 from p_4 and signer revealed y_0^5
2. The verifier computes y_1^2 from y_0^4 and y_0^5 and signer revealed y_1^3
3. The verifier computes y_2^1 from y_1^2 and y_1^3 and signer revealed y_2^0
   Here in the second approach all of the redundant terms that appear in the last approach had been calculated and hence it saves the memory of the verifier from storing all the redundant terms, they are instantly being calculated and use.

The revealed elements as in the last approach(say) are the authentication path i.e. y_0^5, y_1^3 and y_2^0 It had been calculated and found that the second methods take only $log_2 n$ compared to the prior where it takes $2log_2 n$ The method is given by Merkle only needs to save the public verification key Y_d^0 for 2^d messages no need to store all the 2^d verification keys(refer Table.6).

The approach from the leaves of the tree to the root seems to be well established but still, there is one more thing to do, it's quite disturbing that even though the authentication path of p_{i+1} has a big portion of the authentication path for p_i. It will be quite useful and memory and time saving that instead of calculating authentication path for each of the p_i we calculate all the p_i in order staring from p_0 and going up to p_{2^d} and avoid all that path elements which depend on the previous and keep only those path elements that are new to the p_i underuse and develop the path by the approach of recurrence(refer Figure.7).

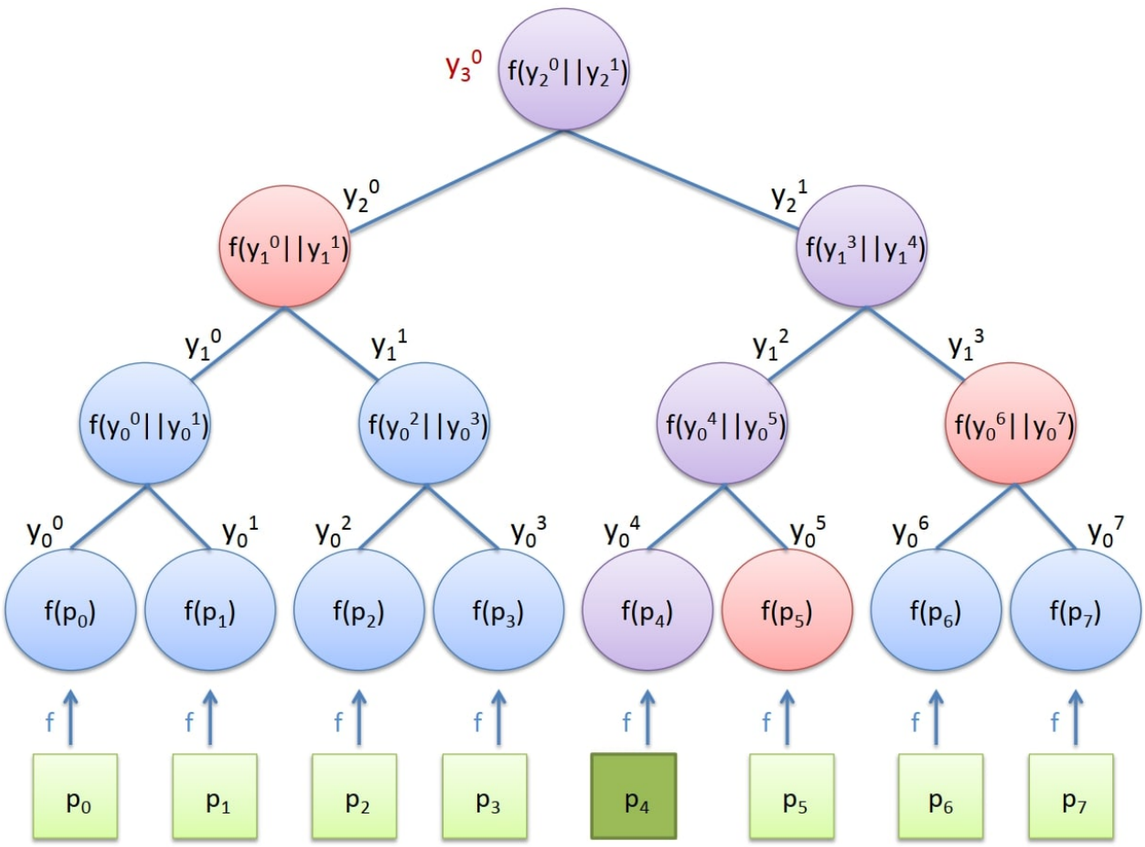

Figure 7: Merkle Static Tree of height 3

Table 6: Authentication table of the tree of height 3

p_i	**Authentication Path**		
p_0	y_2^1	y_1^1	y_0^1
p_1	y_2^1	y_1^1	y_0^0
p_2	y_2^1	y_1^0	y_0^3
p_3	y_2^1	y_1^0	y_0^2
p_4	y_2^0	y_1^3	y_0^5
p_5	y_2^0	y_1^3	y_0^4
p_6	y_2^0	y_1^2	y_0^7
p_7	y_2^0	y_1^2	y_0^6

The black are non-redundant elements and the rest are redundant elements.

2.10.1 Signature generation and verification using the static version of Merkle tree

The above Merkle signature Scheme uses Lamport Scheme and its variants. The average size of the signature developed by the above scheme is roughly about $2n^2 + nlog_2N$ where n is the length of the digest of the one-way hash function, i.e. $h : \{0,1\}^* \rightarrow \{0,1\}^n$ and $N = 2^d$ the total number of messages to be signed and $2n^2$ is caused by the one-time signature scheme and $nlog_2N$ is caused due to the authentication path.

Typically, the value for n and N are 256 and 2^{20}

The basic version of the Merkle Scheme uses the Lamport and its variant but the extended version of it developed by Buchmann, Dahmen and Husling called as, "eXtended Merkle Signature Scheme" or XMSS in short.

2.10.2 eXtended Merkle Signature Scheme or XMSS

The main difference between the prior version and the current version are smaller signatures and collision resilience and the use of Winternitz instead of Lamport Scheme and the main advantage of this is that there is no need to store pk_i instead it can be calculated and compared to give one; in the context of the prior version, this means that instead of storing

Algorithm 8: Signature generation and verification using the static version of Merkle tree

1. **Signature Generation**
   - The signature is developed correspondingly as in an onetime signature scheme, the signature for the i^{th} message is done using the index i, the one-time signature for the i^{th} message developed using its secret key $(sk_i)s_i$, the public key for the i^{th} message (pk_i) and the corresponding Authentication path $(Auth_i)$ using the following relation $\sum(i, s_i, pk_i, Auth_i)$
2. **Signature Verification**The signature is verified also in a very similar fashion as in a one-time signature scheme
   - Verifying that m_i had been signed using pk_i using the one-time signature scheme as described earlier in the paper.
   - Authenticating the public key pi using the authentication path and finally computing Y_d^0 and comparing it with the given values.

the pk_i, it is being computed from the signature and eventually with the help of the computed values the root value is computed i.e. Y_d^0. [7]
This reduces the signature size from $2n^2 + nlog_2N$ to $n^2 + nlog_2N$ the Winternitz parameter i.e. the number of messages to be signed simultaneously, $w \in N$ (set of natural numbers).
After looking at a brief overview of the some of the fundamentals of hash-based signature let defines stateful and stateless hash-based signature scheme.
Stateful hash-based signature scheme: This scheme is identified with the characteristics of maintaining the state which essentially keeps track of the used One-time signature keys so that the keys cannot be used to sign multiple messages. Ex: XMSS, LMS

Stateless hash-based signature scheme: This scheme is identified with the characteristics of not maintaining the state, the signatures are much larger and also computationally much intensive. Ex: SPHINCS

2.11 Advantage of using Hashed based signature scheme

1. Reliable hash function: The ultimate security of the hashed based cryptography lies with the security of the underlying hash functions and much of the hash function approved by NIST are quantum-safe.
2. Standardization: Stateful hash function are the most likely to be standardized very soon by NIST and thus they can be implemented
3. Utilize Currently available hardware: Unlike the rest of the post-quantum cryptosystems Hash-Based Cryptosystem involves the majority of the calculation on the hash function and much of them had been optimized in the architecture.
4. Smaller Key size: The public and private keys have a much smaller size compared to other post-quantum cryptosystems by using Merkle tree the need for storing a large number of verification keys had been reduced to one.
5. Easy to adapt and implement: If there is any security issue with any of the hash-based cryptosystem just change the underlying hash function nothing else to do no need for complex hardware implementations and mathematical computations.

2.12 Comparison between Quantum cryptography and Post Quantum cryptography: Quantum Key Distribution

The two-term quantum cryptography and post-quantum cryptography seem so, much similar yet a considerable amount of difference exists between them.
Quantum Cryptography refers to the use of fundamental laws of Quantum mechanics such as Quantum superposition, quantum entanglement, Heisenberg uncertainty principle, dual

behavior of matter and radiation, de Broglie Hypothesis, and another fundamental principle to deals with the encryption and decryption process and developed security of the system. Quantum Key Distribution (QKD) is the best-known example of the quantum encryption method where the data (the keys) are transferred using photons and the advantage of a photon's no change no-cloning attribute can be used to harnessed security. In this method, the third party cannot get the keys from the photons and if somehow the attacker gets access to the keys through some messages these would ultimately change the state of the photons and the cryptosystem fails to alert the active parties that their data had been altered by some adversary.
Post-Quantum Cryptography refers to the use of a cryptographic algorithm that is believed to be safe from quantum attacks i.e. resistant to Shor's, Grover's algorithm. Post-quantum cryptography is all about building quantum-safe cryptosystems by updating our present knowledge of mathematics and computer science. [6]
Though much of the post-quantum cryptosystem had not received much attention from the cryptographic society except hash-based cryptography whose security is well understood and most likely they would be implemented on a large scale in the coming years.

3 Conclusion

The era of Quantum computing is not very far away, Google had achieved Quantum supremacy already. Several institutions are currently working on Quantum Computing like D-Wave, Google, AT & T, Atos, Honeywell, and many other institutions and companies, so it's no more surprising that within the next few years may be within this decade that we enter into the quantum era. Our classical Cryptosystems are no doubt safe now but we cannot leave our research on post-quantum cryptography and look at the current trend the hash-based cryptography is the most promising compared to the other which involved much complex hardware issues and mathematical stimulation. So, it can be safely stated that instead of searching for other post-quantum cryptosystems, we first focused on what we have now, build more secure hash functions, optimize the architectural design for their implementations, and developed new schemes.

References

[1] Gauthier Umana,Valérie, Post Quantum Cryptography. [Doctoral Thesis], Technical University of Denmark, vol.322, pp.79-87, 2011.

[2] Daniel J. Bernstein,Johannes Buchmann,Erik Dahmen, "Post Quantum Cryptography", Springer, 10, pp.35-43, 2009.

[3] Kumar Sekhar Roy and Hemanta Kumar Kalatia, "A Survey on Post-Quantum Cryptography for Constrained Devices", "International Journal of Applied Engineering Research", Research Area Publication, v.14, no.11, pp.1-7, ISSN:0973-4562, 2019.

[4] Nguyen Thoi Minh Quan, "Intuitive Understanding of Quantum Computation and Post-Quantum Cryptography", arXiv:2003.09019, v.9,

pp.8-9,2000.

[5] Peter W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer", arXiv: quant-ph/9508027v2, v.2, pp.5-23,1996.

[6] Lily Chen; Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner and Daniel Smith-Tone, "Report on Post-Quantum Cryptography",NISTIR 8105, v.2,DOI=http://dx.doi.org/10.6028/NIST.IR.8105, pp.1-7,2016.

[7] Andreas Hülsing, Stefan-Lukas Gazdag, Denis Butin and Johannes Buchmann, "Hash-based Signatures: An Outline for a New Standard", NIST,

DOI=https://csrc.nist.gov/csrc/media/events/workshop-on-cybersecurity-in-a-post-quantum-world/documents/papers/session5-hulsing-paper.pdf, pp.1-3.

[8] Johannes Buchmann and Jintai Ding, Editors: Cincinnati, “Hash-based Signatures: An Outline for a New Standard, Post-Quantum Cryptography, Second International Workshop, PQCrypto 2008, OH, USA”, Lecture Notes in Computer Science, v.5299,Springer.

[9] Noson S.Yanofsky and Mirco A. Mannuci, “Quantum Computing for Computer Scientists”,Cambridge University Press,pp.204-209,ISBN 978-0-521-879965,2008

[10] N.David Mermin, “Quantum Computer Science An Introduction”,Cambridge University Press,pp.63-69,ISBN-13 978-0-521-87658-2,2007

[11] Pawan Kr Pradhan, Sayan Rakshit, Sujoy Dutta, “Lattice-Based Cryptography, Its Applications, Areas of Interest and Future Scope”, Third International Conference on Computing Methodologies and Communication, ISBN-978-1-5386-7808-4,2019

[12] N. Sendrier, “Code-Based Cryptography: State of the Art and Perspectives”, in IEEE Security Privacy, vol. 15, no. 4, pp. 44-50, 2017, doi: 10.1109/MSP.2017.3151345.

[13] J. Ding and A. Petzoldt,“Current State of Multivariate Cryptography” in IEEE Security Privacy, vol. 15, no. 4, pp. 28-36, 2017, doi: 10.1109/MSP.2017.3151328.

[14] G. Arun and V. Mishra,“A review on quantum computing and communication”, 2014 2nd International Conference on Emerging Technology Trends in Electronics, Communication and Networking, Surat, 2014, pp. 1-5, doi: 10.1109/ET2ECN.2014.7044953.

[15] L. O. Mailloux, C. D. Lewis II, C. Riggs and M. R. Grimaila, “Post-Quantum Cryptography: What Advancements in Quantum Computing Mean for IT Professionals”, in IT Professional, vol. 18, no. 5, pp. 42-47, Sept.-Oct. 2016, doi: 10.1109/MITP.2016.77.

[16] F. Borges, P. R. Reis and D. Pereira, “A Comparison of Security and its Performance for Key Agreements in Post-Quantum Cryptography”, in IEEE Access, vol. 8, pp. 142413-142422, 2020, doi: 10.1109/ACCESS.2020.3013250.

[17] Y. Gorbenko, I. Svatovskiy and O. Shevtsov, “Post-quantum message authentication cryptography based on error-correcting codes”, 2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC ST), Kharkiv, 2016, pp. 51-54, doi: 10.1109/INFOCOMMST.2016.7905333.

[18] PhysRevA.101.062310, title = Quantum singular value decomposer, author = Bravo-Prieto, Carlos and García-Martín, Diego and Latorre, José I., journal = Phys. Rev. A, volume = 101, issue = 6, pages = 062310, numpages = 6, year = 2020, month = Jun, publisher = American Physical Society, doi = 10.1103/PhysRevA.101.062310, url = https://link.aps.org/doi/10.1103/PhysRevA.101.062310