

**Original Article**

# **POST-QUANTUM THREATS TO SSH PROTOCOL: A PUBLIC KEY SHARING PERSPECTIVE**

***Youssef Ahmed El-Mokhtar***

Faculty of Sciences of Rabat, University

Mohammed V, Rabat, Morocco.

DOI: <https://doi.org/10.5281/zenodo.17100256>

**ABSTRACT:** The security of traditional cryptographic schemes is based on mathematical puzzles that cannot be cracked by current computers. However, the rapid development of quantum computers has the potential to greatly reduce the time and resources required to crack these encryption schemes. Although true quantum supremacy may still be years away, it is imperative to adopt anti-quantum algorithms proactively. This preemptive approach aims to thwart "catch first, decrypt later" attacks, wherein attackers intercept and store encrypted data with the intent of decrypting it once quantum computing becomes sufficiently advanced. This threat is particularly critical for protocols like Secure Shell (SSH), which is widely used for secure communication over unsecured networks. In this paper, we propose an innovative approach to enhance the security of public-key distribution within the SSH protocol. Our method integrates quantum-resistant algorithms to ensure that even with the advent of quantum computing, the confidentiality and integrity of SSH sessions are maintained. We emphasize the importance of using cryptographic protocols that operate over open channels, which, while not necessarily confidential, must be authenticated to prevent tampering. In such scenarios, attackers may be able to intercept and even extract information, but they should not be able to alter the data.

**KEYWORDS:** SSH; BB84; Encryption; Quantum.

## **INTRODUCTION**

Equipment and servers, making remote login technology a systems to resist quantum attacks. Future quantum computers could potentially decrypt today's cipher text data, posing a significant threat to information security. Lattice-based cryptography has garnered considerable attention in recent years as a promising defense against such threats [1]. The security of these encryption mechanisms hinges on the difficulty of solving certain mathematical problems, known as security hypotheses. If the underlying mathematical problem is cracked, the corresponding encryption mechanism is also compromised [2]. Thus, a critical challenge in designing encryption mechanisms is to identify suitable mathematical problems that are both sufficiently difficult and thoroughly

## **Original Article**

studied. Many of these mathematical problems are rooted in number theory, with some of the most widely used problems being factoring and the discrete logarithm problem [3-5]. Concurrently, the evolution of internet technology and the expansion of network scales have heightened the demand for secure remote login solutions. Many enterprises and organizations rely on remote login services to manage network significant security risks to users [7]. The Secure Shell (SSH) protocol emerged as a solution, encrypting transmitted data to ensure secure communication. Currently, SSH utilizes symmetric encryption algorithms for data transmission. This process requires both parties to securely negotiate a shared key, typically generated by the Diffie-Hellman (DH) algorithm [8]. The security of the DH algorithm is based on the difficulty of computing discrete logarithms [9]. However, with recent advancements in quantum theory, researchers have developed quantum algorithms capable of solving discrete logarithms in polynomial time [10]. This development undermines the security of the DH algorithm and, consequently, the SSH protocol, presenting significant challenges and threats. To address the quantum algorithm threat to the SSH protocol, this paper delves into the relevant aspects of the SSH protocol and proposes an improvement scheme using the BB84 key exchange protocol [11]. The BB84 protocol, a quantum key distribution method, offers a quantum-resistant alternative for secure key exchange [12]. In the subsequent sections, we will explore the advent of quantum computing, there is a highly relevant research topic [6]. Initially, remote services increasing necessity to enhance current encryption like Telnet and FTP transmitted data in plaintext, posing Explore related work on our method, examining the functionalities of SSH and BB84. Finally, we will present our proposed solution in this context, demonstrating how it can enhance the security of SSH against quantum threats [13-16].

## **RELATED WORKS**

Several studies have proposed hybrid cryptographic methods that integrate classical and quantum cryptography. Beginning with [17], this study implemented a secure approach that effectively meets quality of service requirements across diverse channel conditions (free, loaded, and congested). It leverages the SSL/TLS protocol's robustness and flexibility, utilizing encryption keys from its cipher suite list, particularly focusing on the Advanced Encryption Standard (AES) with its various key sizes (128, 192, 256). AES is chosen for its resistance against classical attacks (such as differential and linear attacks) and its efficiency compared to other protocols like DES and 3DES. [18] Builds upon improvements in Hill's encryption to enhance Vigenere's algorithm by addressing its weaknesses, such as key size detectability and inability to encrypt identical letters in different blocks. By combining these approaches, a hybrid algorithm is formed that offers reliability and resistance against a wide range of attacks, including statistical attacks. [19] Introduces a password security algorithm for online authentication where user passwords are hashed using the SHA3 function before storage in a remote database. The system adds random rotations to the hashed passwords to prevent repetitive database attacks. The stored hash is then retrieved for comparison during user authentication. [20] Describes an enhanced method for securely storing passwords using MD5 hashing combined with predefined transformations before database storage. Additionally, [21] discusses a new SSL/TLS extension, Quantum SSL (QSSL), designed to simplify the integration of quantum key distribution (QKD) into existing Internet security infrastructures. QSSL aims to enable practical deployment of quantum cryptography-based security applications, offering secure encryption and unconditionally secure authentication based on the QKD BB84 protocol. [22] Examines networks composed of multiple nodes capable of executing security protocols protected by physical laws. It explores the integration of QKD protocols into network security infrastructures, highlighting the need for modeling and simulation to

## Original Article

understand Quantum Key Distribution Networks (QKDNs). The paper emphasizes the critical issue of unconditionally secure public channel authentication in QKD. [23] Presents a quantum and hybrid protocol addressing network computation security, enhancing two-party and multiparty computations in geographically distributed networks using quantum key distribution and 3DES encryption over a quantum channel within cloud infrastructures. [24] Proposes a quantum-based solution to security challenges, ensuring data confidentiality and authenticity by replacing conventional SSL or SET connections with quantum cryptographic systems. This model offers a robust security framework for online banking and other applications. This collection of studies underscores the ongoing evolution and integration of quantum and classical cryptography to address diverse security challenges in networked environments.

### SSH PRINCIPLES

SSH uses a combination of Public Key and Secret Key. The Public Key is used to transmit the Secret Key and perform identity authentication between the client and the server before establishing a secure channel. Encrypt and decrypt (table1):

Table 1. Different keys in SSH.

Name	The life cycle	Create	Type	Describe
Host Key	Persistence	Server	Public Key	Authenticate to the server
User Key	Persistence	user	Public Key	Authenticate clients (users)
Server Key	Default is 1 hour	Server	Public Key	Used to encrypt Session Key (only SSH-1 protocol has it, SSH-2 has enhanced it, here Server Key is used as a concept to facilitate description in the process)
Session Key	client	Session	Secret Key	Used to encrypt transmitted data

#### Key Features of SSH

- Encryption: Avoid data leakage.
- Integrity of communication: avoid data tampering, and send or receive address masquerading.
- (check whether data has been tampered with and whether the data is from the sender and not the attacker)

SSH-2 implements this function through MD5 and SHA-1, SSH-1 uses CRC-32 [25].

- Authentication: Identify data sender and receiver identities the client verifies the identity of the SSH server: Prevent attackers from spoofing the identity of the SSH server, avoid intermediary attacks and redirect requests; OpenSSH stores the host name and host key in know-hosts. Authenticate the identity of the server, verify the identity of the requester on the server: provide a less secure user password method and a more secure per-user public-key signatures; in addition, SSH also supports integration with third-party security service systems, such as Kerberos, etc [26].

- Authorization: User Access Control.

## Original Article

- Forwarding or tunneling to encrypt other TCP/IP- based sessions it can provide communication security for Telnet, FTP, etc. through SSH, and supports three types of forwarding operations: Port Forwarding; X Forwarding; Agent Forwarding [27].

SSH is the predominant protocol for remotely controlling computers within a network. When initiating an SSH session between devices, several critical steps ensure secure and authenticated communication:

- **Setting up an Encrypted Channel:** The first step involves establishing an encrypted channel between the client and the server. This encrypted channel provides a secure pathway for exchanging messages, ensuring that all data transmitted between the devices remains confidential and protected from eavesdropping or tampering [28].
- **Data Integrity Verification:** Once the encrypted channel is established, SSH verifies the integrity of data sent by the client. This process ensures that the data has not been altered or corrupted during transmission, maintaining the accuracy and reliability of the information exchanged [29].
- **Client Authentication:** Following data integrity checks, SSH proceeds with client authentication. Authentication confirms the identity of the client attempting to access the remote host. SSH supports various authentication methods, including password- based authentication and public-key cryptography. This step is crucial for ensuring that only authorized clients can access the server, thereby preventing unauthorized access and safeguarding sensitive data [30].

By following these three essential steps—setting up an encrypted channel, verifying data integrity, and authenticating the client—SSH facilitates secure and reliable communication between computers on a network. This approach allows users to confidently share confidential information and ensures that access to network resources is controlled and secure.

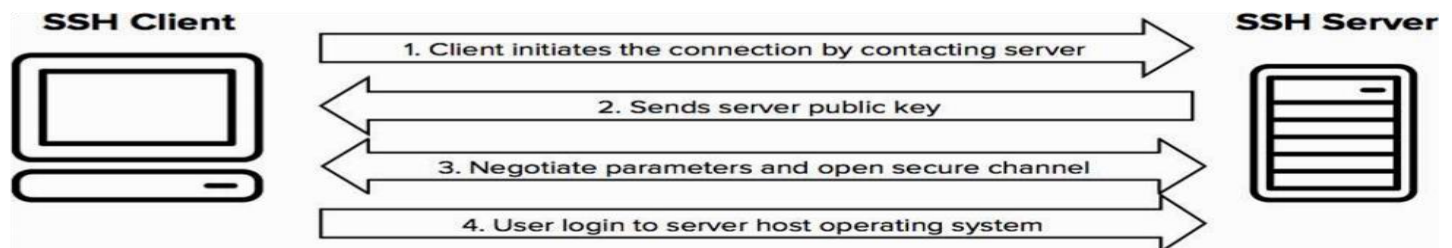


Figure 1. Establishing an SSH connection

THE QUANTUM KEY DISTRIBUTION PROTOCOL-BB84. Several quantum cryptography techniques are based on the conveyance of information utilizing single-photon state coding to protect sensitive information. Charles Henry Bennett suggested and Gilles Brassard) in 1984 that a new quantum key distribution technique be developed. Similar to previous protocols, its goal is to generate a fresh session key that may be used in traditional symmetric cryptography to encrypt data. But one distinctive aspect of the protocol is that it uses certain quantum physics features to guarantee that an intruder does not intercept the received key. This is a very important feature since it protects the key from being intercepted. The protocol assumes that an attacker's entry into the protocol can be detected until the attacker has complete control over all communication channels, including the ability to read and write data [31]. The procedure is divided into three sections:

- Photons are sent and received using a quantum communication channel.
- Communication between the transmitter and receiver about the analyzers employed.

## Original Article

- Communication between the receiver and transmitter regarding the coincidence between the chosen analyzers and the starting polarizations.

The quantum states of a system can be described as follows:

$$|0_x\rangle = \frac{1}{\sqrt{2}}(|0_+\rangle + |1_+\rangle), \quad |1_x\rangle = \frac{1}{\sqrt{2}}(|0_+\rangle - |1_+\rangle),$$

Here the states  $|0_+\rangle$  and  $|1_+\rangle$  encode the values "0" and "1" in the "+" base;  $|0_x\rangle$  and  $|1_x\rangle$  encode the same values in the "x" base. The bases are rotated by  $45^\circ$  in relation to each other (Figure 2) [32, 33].



Figure 2. Polarization states of the photons used in protocol BB84.

A. THE STEPS IN THE CONSTRUCTION OF THE KEYS: 1) The transmitter randomly chooses one of the bases. Then in the base randomly chooses one of the states corresponding to 0 or 1 and sends photons (figure 3):

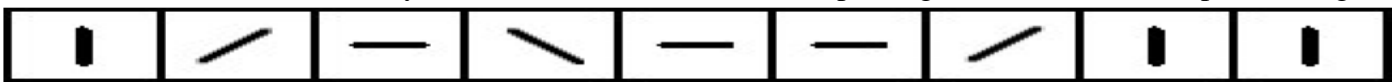


Figure 3. Photons with different polarization.

2) The receiver chooses randomly and independently of the transmitter for each incoming photon: straight (+) or diagonal (x) base (figure 4):

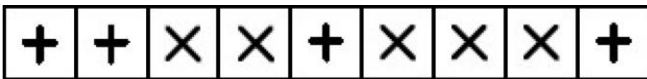


Figure 4. Selected measurement type.

The recipient then records the measurements as shown below:



Figure 5. Measurement results

3) The receiver, via an open and publicly accessible communication channel, reports what type of information the following measurements were used for each photon, i.e. which baseline was selected, but the results of the measurements remain secret;

4) The sender informs the receiver via the open public channel which measurements have been selected according to Alice's initial baseline (Figure 6):



Figure 6. Evolution of the frequency of correct measurements

5) Users then leave only those cases in which the selected bases coincide. These cases are converted into bits (0 and 1), thus obtaining the key (Figure 7):







vertical bar			diagonal backslash	horizontal bar		diagonal slash		vertical bar
1			1	0		0		1

Figure 7. Obtaining a key sequence from correct measurements

## Original Article

The number of cases in which the selected bases coincide will be, on average, half the length of the original sequence, i.e.,  $n=1/2$  on average half the length of the original sequence, i.e. (an example for determining the number of photons taken by Bob is shown in Table 2).

Table 2. Quantum key generation using the BB84 protocol

The binary signal from the transmitter	0	1	0	1
The polarization code of the transmitter				
Detecting the recipient				
The recipient's binary signal	0	1	?	?

Thus, the transmission of Bob's key, in the absence of interference and distortion, will result in an average of 50% of the photons being correctly recorded. However, there are no perfect communication channels and additional error recovery and secrecy enhancement procedures must be applied to form a secret key. At the same time, for a part of the user's bit sequence where the bases coincide, the values of the bits are randomly revealed and compared through an open public communication channel. In addition, the revealed bits are discarded. An ideal (noise-free) quantum channel is sufficient to detect a mismatch in a single disclosed position to detect an intruder. It is impossible to distinguish between errors caused by noise and those caused by an intruder in a real-world situation. It is known that if the  $QBER \leq 11\%$ , users of an undisclosed sequence, after error correction by an open public communication channel and secrecy enhancement, can extract a secret key that will be the same for them. The key obtained before additional operations on the sequence is called the "raw" key [34].

## V. THE PROPOSED APPROACH

In order to achieve the transition to quantum-secure computing, security protocols such as SSH, VPN, IPSec, SSLTLS, etc. also need to be upgraded. These protocols need to be combined with existing protocols, and an additional layer needs to be introduced to establish secure communication to deal with countermeasures. Protection from quantum attacks. This change has implications for asymmetric encryption and key generation algorithms, requiring an increase in the key size of symmetric cryptographic algorithms. As a result, performance and bandwidth are also impacted. Hardware vendors will also need to upgrade their hardware to align and transition with these new algorithms.

SSH employs both asymmetric and symmetric encryption methods. This is how it works.

- To begin, asymmetric encryption is used to exchange a secret symmetric encryption key privately.
- Then, we use the symmetric encryption key to encrypt the exchanges.

Why not use only asymmetric encryption since it is too reliable? This would be possible, but there is a flaw: asymmetric encryption requires too many processor resources. Asymmetric encryption is 100 to 1,000 times slower than symmetric encryption. Computers, therefore, exchange the symmetric encryption key securely (thanks to asymmetric encryption) and can then communicate faster by using symmetric encryption all the time. The system architecture of a typical hybrid SSH application is shown in Figure 8. The figure shows that SSH sits between the applications layer protocols and TCP. On transmission, SSH accepts data traffic from the application

## Original Article

layer and adds the required security protection before transmitting it to the lower layers. BB84 is used as the QC (Quantum Cryptography) protocol in this architecture. SSH uses the network to send only traffic corresponding to the classic public channel exchanges required to implement BB84. The latter type of traffic is transmitted during the SSH handshake (Figure 8) and must be authenticated to thwart man-in-the-middle attacks. To avoid man-in-the-middle attacks on the QKD protocol.

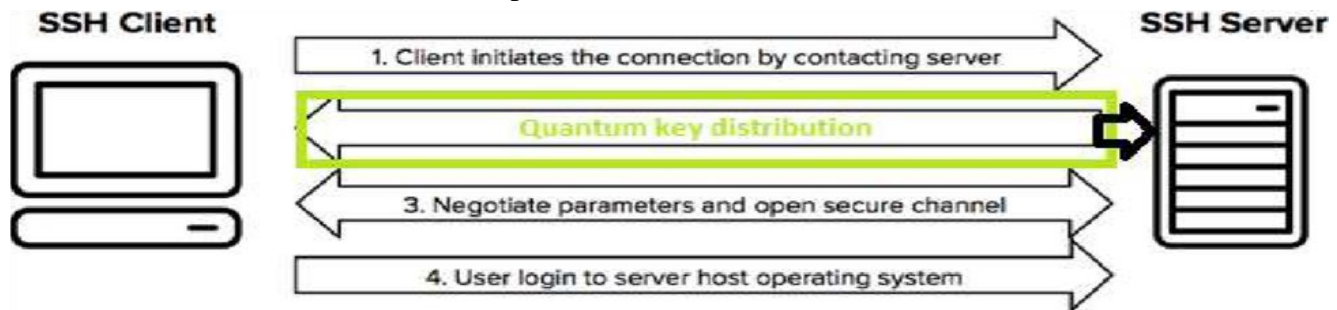


Figure 8. Proposed change

For more precision, our proposal consists in modifying the exchange stage using the classical symmetrical key, and to pass by a quantum key based on BB84 as illustrated below.

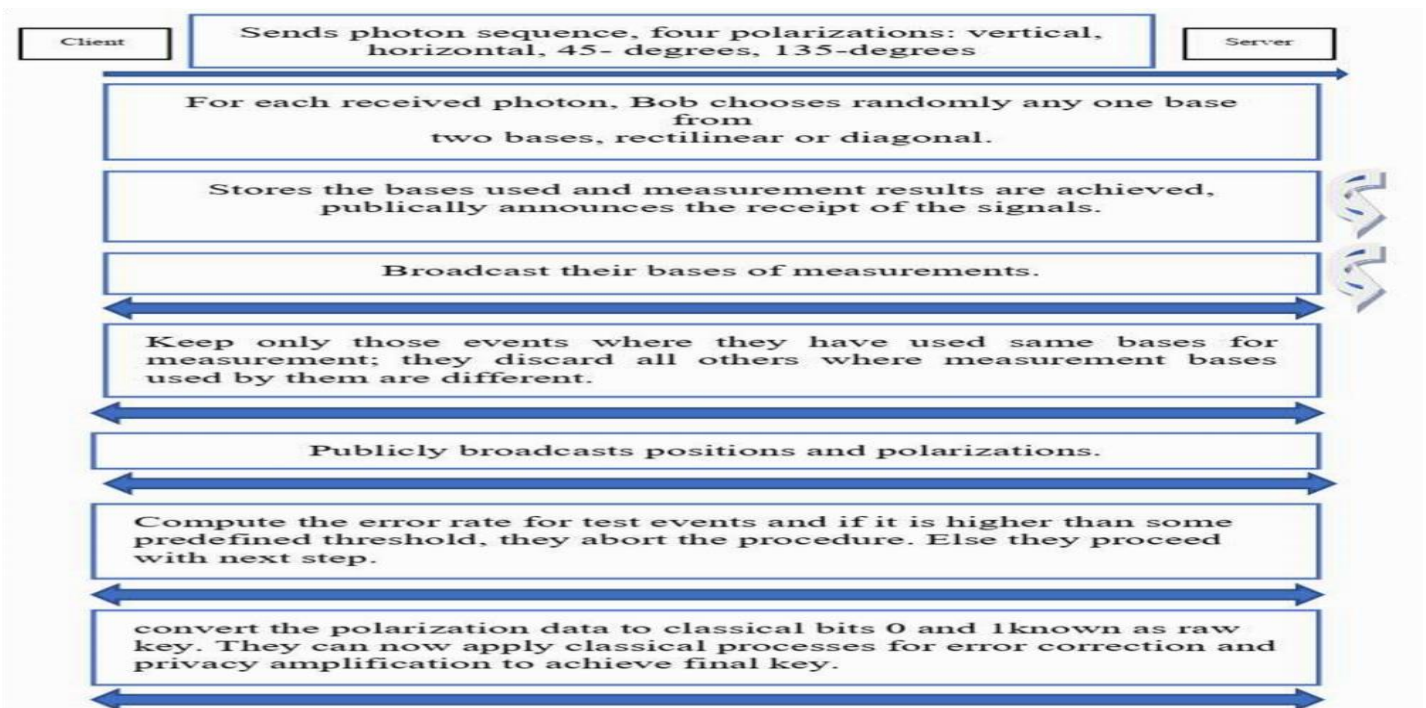


Figure 9. Generation of a quantum symmetric key.

The sequence of the different phases of the quantum key distribution between the client and the server will be as follows: both will start a classical communication until the key generation phase. The interlocutors will have quantum bits that they measure using different polarization angles. To generate the secret key, their measurement results must be identical, and in the process, some bits are eliminated during the filtering process. The error rate is calculated after filtering, and if the error rate is higher than a predefined threshold, the key distribution process

## Original Article

will be aborted. Aborted. This may be due to the presence of a spy or a high noise level. If the error rate is below the threshold, privacy amplification is performed in order to reduce the information gain by the hacker. Privacy amplification is performed to reduce information gain by the eavesdropper further (as illustrated below).

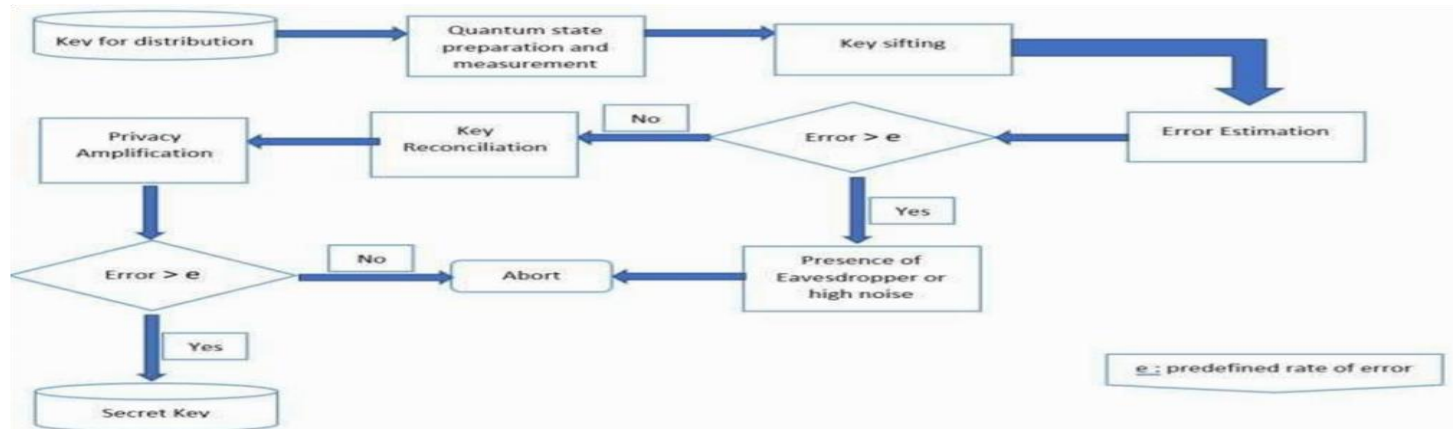


Figure 10. The various stages of quantum public key distribution.

## VI. EXPERIMENTATION AND DISCUSSION

### QUANTUM PUBLIC KEY GENERATIONS

The key, as previously stated, is digital identity. It is a one-of-a-kind binary data string that means "I am your client." Furthermore, the SSH client can use cryptographic magic to prove that its key is genuine to an SSH server. All of this is usually accomplished through four major steps:

1. The client initiates the conversation: "Hey, server, I'd like to SSH into an account on your system, specifically the account of user X."
2. In response, the server says: "Well, perhaps. First and foremost, I challenge you to prove your identity!" Furthermore, the server sends data to the client, known as a challenge.
3. According to the client: "I'll take your challenge. This is proof of my identity. I created it by mathematically combining your challenge and my private key." This response to the server is referred to as an authenticator.
4. According to the server, "Thank you for providing the authenticator. I will now look into account X to see if you can log in." The server specifically checks X's public keys to see if the authenticator "matches" any public keys. To see if the authenticator "matches" any of them, go to X. (Another cryptographic operation is "matching.") If so, the server will say, "OK, enter!" Authentication will fail otherwise. We generate a quantum key (table 3) and then summarize the classical and the quantum method.

Table 3. Generation of a quantum public key with BB84.

	0	1	0	1	0	1	1	1	0	1	0	1	0	1	1	1
SSH Basis Client	+	X	+	X	+	X	X	+	+	X	+	X	+	X	X	+
Polarisation du client SSH	↑	/	→	\	↑	\	/	→	↑	/	→	\	↑	\	/	→
SSH Basis Server	+	+	X	X	X	+	X	+	+	+	X	X	X	+	X	+
SSH server polarization	↑	→	\	\	/	→	/	→	↑	→	\	\	/	→	/	→
<b>Public discussion of measurements</b>																
<b>The public key</b>	0			1			1	1	0			1			1	1

## RESISTANCE TO ATTACK

### Man-in-the-Middle

## **Original Article**

In reality, the system is vulnerable to a man-in-the-middle assault. Future connections to this server remain secure, however, given you were not spoofed at the time of your first connection. This is true even if the server host key is taken. The host of the server has not been stolen. The second layer of security provided by the approach is increased authentication and data sharing via the use of a quantum key, since the password method is insecure. The public key and host-based authentication, on the other hand, are both resistant to MITM attacks. By just witnessing key exchanges, the attacker will not be able to determine the session key; instead, he will need to launch an active assault. He engages in many exchanges with each partner, resulting in the acquisition of multiple keys. A number of different exchanges are made with each party in order to receive their keys with the client and server. If this occurs, the key exchange on either side of the key exchange is intended to delete the session identification numbers on both sides of the key exchange. This is true for both SSH-1 and SSH2. There are two separate session IDs on each side. When a client gives a digital signature for public key or host-based authentication, the digital signature contains the identification of the data that was signed by the client. Consequently, the attacker is unable to simply transfer the authenticator given by the client to the server and has no method of compelling the client to sign any other session identifier. IP and TCP Attacks Because SSH is built on top of TCP, it is susceptible to attacks that take use of TCP and IP weaknesses. The assurances of confidentiality, integrity, and authentication provided by our technique significantly reduce this susceptibility to denial-of-service assaults. Because it is robust to network challenges such as congestion and connection loss, TCP/IP is used for data transmission. If a hacker manages to take down a router, IP may get around it. A malicious opponent introducing false network packets was not expected to be able to defeat it. It is not possible to determine the source of TCP or IP control messages.

Consequently, TCP/IP features various inherent weaknesses that may be exploited, such as SYN flooding, which is a kind of packet flooding. "Synchronize" is an abbreviation for the TCP packet property SYN, which means "synchronize." In this particular instance, it is the initial packet transmitted to commence the formation of a TCP connection. Due of the frequency with which this packet is received, the receiver must regularly consume resources in order to prepare for the next connection. If an attacker transmits a large number of these packets, the recipient TCP stack may become overburdened and fail to establish a connection. In the case of a TCP connection, a RST packet may be sent by any side at any moment, and the connection will be instantly ended as a result. RST packets may be instantly injected into a network, causing any target TCP connection to be instantaneously terminated and disconnected.

### **Traffic Analysis**

A hacker who cannot read your network traffic might nevertheless gain valuable information by merely watching it and recording the volume of data sent, as well as the source and destination addresses, as well as the date and time of transmission. In the case of another organization, an unexpected surge in traffic might signal the beginning of a commercial relationship. Traffic patterns may also be used to predict when backups are most likely to occur or when denial of service assaults are most likely to occur. It is possible that a lengthy period of quiet on an SSH connection from a system administrator's desktop indicates that he has taken some action. Whenever a system administrator indicates that he or she has left the facility, it is a great time to break in, either electronically or physically, and take advantage of the situation. Attacks against SSH that are based on traffic analysis are not tolerated. Because SSH connections are often addressed to a well-known port, our approach is very rapid to identify SSH connections. Hide the results of traffic analysis. To avoid activity correlation, an SSH

## Original Article

implementation might broadcast random traffic on an idle connection in order to prevent activity correlation from occurring.

### Comparison

In our comparison between the standard SSH protocol and the one described in this article; we can see that ours is more resistant to a variety of assaults. The conventional SSH protocol does not provide protection against attacks that interrupt or impede the formation of TCP connections, which began with IP and TCP assaults. SSH's encryption and host authentication, on the other hand, are effective against attacks employing routing in order to read sensitive information or divert a connection to a compromised server, as seen in the example above. Attacks that reroute or change TCP data, on the other hand, fail because SSH identifies them, but they also fail because SSH does not detect them, resulting in the termination of the SSH connection. Due to the fact that all communication is reviewed during the connection formation phase (and in particular during the quantum key creation phase) when using Quantum SSH, the protocol is immune to this vulnerability. The cracking of passwords is another another potentially deadly assault. SSH greatly increases password security by encrypting passwords as they are sent via a network connection. A password, on the other hand, is a weak type of authentication that should be used with extreme care. It is difficult to guess a decent password that is both memorable and not evident to others when they are not provided. The BB84 protocol is successful in terms of encryption since it offers a number of different techniques for encrypting data that is received across the network. The consequence is that it is possible to construct a cryptographic layer that is resistant to this kind of assault. The following table indicates the degree to which each approach is resistant to the most well-known assaults on the Internet.

Table 4. Classical VS quantum SSH resistance against different attacks.

	Man-in-the-Middle	IP and TCP Attacks	Traffic Analysis	Password Cracking	Covert Channels
Classic SSH	✓	✗	✓	✗	✓
Quantum SSH	✓	✓	✓	✓	✓

## VII. Conclusion

This research has enabled us to proceed to a more critical stage, exploiting the different cryptographic approaches to provide a modern and hybrid cryptographic method that allows the SSH protocol to implement a robust and resistant security layer against different types of attacks. Today quantum technology is not fully deployed on the market, but this does not prevent us from using its robustness to improve the classical solutions available in the field. In this regard, we have combined quantum technology (BB84) to improve the efficiency and security of a classical cryptographic approach (SSH). In terms of security. Vice versa, many proven concepts of classical cryptography, such as one-time buffer, multiparty cryptography, delegated computing, and homographic encryption, can be used to implement quantum fundamentals.

## References

D. J. Bernstein, J. Buchmann, E. Dahmen, (Eds.), Post-Quantum Cryptography, Springer, 2009.

## **Original Article**

- O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM*, vol. 56, issue 6, pp. 1-40, 2009.
- B. A. Buhari, A. A. Obiniyi, "Web applications login authentication scheme using hybrid cryptography with user anonymity," *International Journal of Information Engineering and Electronic Business (IJIEEB)*, vol. 14, no. 5, pp. 42-50, 2022.
- N. Koblitz, A. J. Menezes, "A riddle wrapped in an enigma," *IEEE Security & Privacy*, vol. 14, issue 6, pp. 34-42, 2016.
- M. M. Samy, W. R. Anis, A. A. Abdel-Hafez, H. D. Eldemerdash, "An optimized protocol of m2m authentication for Internet of Things (IoT)," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 13, no. 2, pp. 29-38, 2021.
- P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, issue 5, pp. 1484-1509, 1997.
- D. Jao, L. De Feo, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," *Post-Quantum Cryptography*, pp. 19-34, 2011.
- R. L. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, 21(2), 120-126, 1978.
- W. Diffie, M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, issue 6, pp. 644-654, 1976.