תמונה שמכילה גופן, גרפיקה, טקסט, לוגו

התיאור נוצר באופן אוטומטי

Software Engineering Department  
ORT Braude College

Capstone Project Phase A – 61998

**Quantum Secure Socket Layer (QSSL)**

**24-2-R-17**

**Submitters:**

**Raz Tibi – Raz.Tibi@e.braude.ac.il  
Matan Czuckerman - Matan.Czuckermann@e.braude.ac.il**

**Supervisor:**

**Ronen Zilber**

**Table of Contents**

1. **Abstract**
2. **Introduction**
3. **Background**
   1. **Introduction to Quantum Computers**
      1. **Qubits (Quantum Bits)**
      2. **Superposition**
      3. **Entanglement**
      4. **Quantum Interference**
      5. **Quantum Gates and Circuits**
      6. **Quantum Error Correction**
4. **Abstract**

The imminent rise of quantum computing presents a critical threat to current cryptographic systems, particularly those underpinning Secure Socket Layer (SSL), Transport Layer Security (TLS), and other protocols relying on RSA-based encryption and Diffie-Hellman key exchange. This article addresses the severe vulnerability of these widely-deployed security protocols to quantum attacks. Specifically, Shor's algorithm, a quantum algorithm designed for integer factorization and discrete logarithm problems, poses a significant risk by rendering RSA and Diffie-Hellman protocols insecure. As quantum computers advance, the integrity of global digital communications is at risk, demanding urgent action to develop quantum-resistant encryption methods.

We introduce Quantum (Safe) SSL (QSSL), a novel approach to secure communication protocols designed to withstand attacks from both classical and quantum computers. QSSL incorporates post-quantum cryptographic algorithms that are currently believed to be resistant to quantum computational methods, including Shor's algorithm. This research outlines the theoretical framework of QSSL, discusses its implementation challenges, and evaluates its potential to safeguard digital communications in the post-quantum era.

By proactively addressing the quantum threat to SSL, TLS, RSA-based protocols, and Diffie-Hellman key exchanges, QSSL aims to ensure the continued security and privacy of internet communications, financial transactions, and sensitive data exchanges in a world where quantum computers become a reality. This work contributes to the ongoing efforts in cryptography and cybersecurity to prepare for the quantum computing revolution and its implications for global information security

1. **Introduction**

The **Quantum Secure Socket Layer (QSSL)** project is a collaborative effort aimed at addressing the security challenges posed by **quantum computers**. As quantum computing technology advances, traditional cryptographic algorithms become vulnerable to attacks due to their reliance on mathematical problems that can be efficiently solved by quantum computers. The **QSSL** project seeks to develop and promote **quantum-resistant cryptography** to safeguard sensitive data against future quantum threats.

Need to add here the tools and things we use at this project

1. **Background**
   1. **Introduction to Quantum Computers**

Quantum computers operate on the principles of quantum mechanics, which allows them to perform complex calculations much faster than classical computers. Classic physics cannot explain the operation of these quantum devices, and a scalable quantum computer could perform some calculations exponentially faster than any modern computer.  
In particular, a large-scale quantum computer could break widely used encryption schemes.

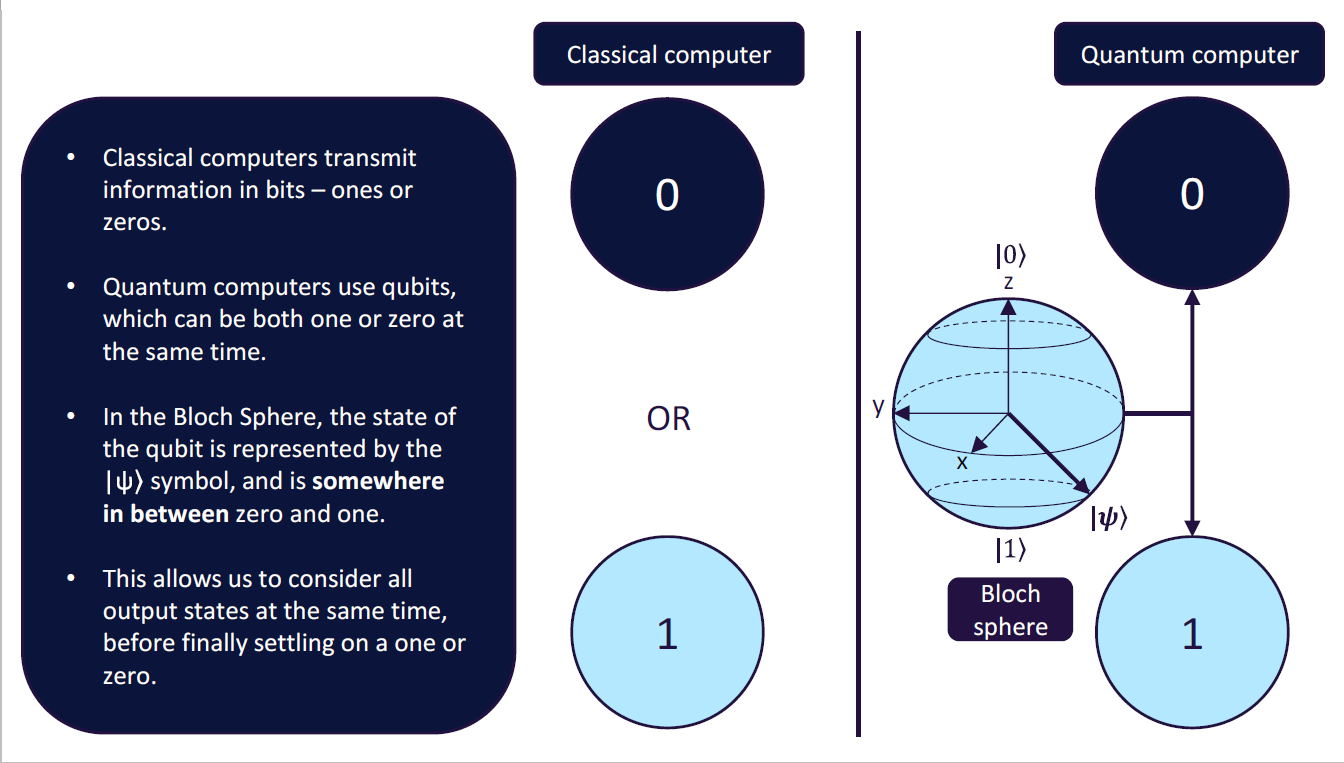
* + 1. **Qubits (Quantum Bits)**

Qubits are the fundamental unit of information in quantum computing, analogous to classical bits. [Unlike classical bits that are either **0** or **1**, qubits](https://www.technologyreview.com/2019/01/29/66141/what-is-quantum-computing/) can represent numerous possible combinations of **1** and **0** at the same time. This ability to simultaneously be in multiple states is called **superposition**. To put qubits into superposition, researchers manipulate them using precision lasers or microwave beams.

* + - 1. **Physical representation**

Qubits can be implemented using various quantum systems, such as:

* + - Superconducting circuits (which isolate an electrical current by eliminating electrical resistance)
    - Trapped ions (which confine a single atomic particle using electromagnetic fields)
    - Photons
    - Quantum dots
    - Nitrogen-vacancy centers in diamond.
      1. **State representation**  
         The state of a qubit is typically represented using Dirac notation, e.g., |0⟩, |1⟩, or a superposition α|0⟩ + β|1⟩, where α and β are complex numbers satisfying |α|^2 + |β|^2 = 1.
      2. **Measurement**When measured, a qubit collapses to either |0⟩ or |1⟩ with probabilities determined by α and β.



*Figure 1*

* + 1. **Superposition**  
       Superposition is a fundamental principle of quantum mechanics that allows qubits to exist in multiple states simultaneously.   
       Mathematical representation: A qubit in superposition is represented as α|0⟩ + β|1⟩. Time frame where qubit remains stable and uncorrupted by external noise is called **coherence.**  Qubits are represented by bits in a superposition state vector – described using complex numbers. When measured or based on certain interactions with the environment, wave function collapses and qubit is transformed into a classical bit based on probabilities, Affect of collapse is called **decoherence**.

**Implications**  
This property allows quantum computers to perform parallel computations on exponentially many states simultaneously.

**Interference**  
Superposition states can interfere with each other, leading to constructive or destructive interference in quantum algorithms.

* + 1. **Entanglement**  
       Entanglement is a quantum phenomenon where two or more qubits become correlated in such a way that the quantum state of each qubit cannot be described independently, no matter the distance between them.This correlation can be used while particle is in quantum superposition state – **coherence**.
       1. **Applications**  
          Entanglement is a crucial resource for quantum teleportation, superdense coding, and quantum cryptography.
* **Quantum teleportation** is a technique for transferring quantum information from one location to another without physically transmitting the quantum state itself.
* **Superdense coding** is a quantum communication protocol that allows the transmission of two classical bits of information by sending only one qubit.  
  + - 1. **Measurement**Measuring one entangled qubit instantly affects the state of its entangled partner, regardless of distance.
    1. **Key Aspects of Quantum Interference**

Interference in quantum mechanics is analogous to wave interference in classical physics, but it occurs with probability amplitudes rather than physical waves. It's a direct consequence of the superposition principle and is key to many quantum algorithms.

* + - 1. **Superposition and Amplitude**

Quantum states are described by complex-valued amplitudes.  
The probability of measuring a particular outcome is the square of the magnitude of its amplitude.  
In superposition, multiple basis states coexist with different amplitudes.

* + - 1. **Constructive and Destructive Interference**

Constructive interference occurs when amplitudes add up, increasing the probability of a particular outcome.  
Destructive interference occurs when amplitudes cancel out, decreasing the probability of an outcome.

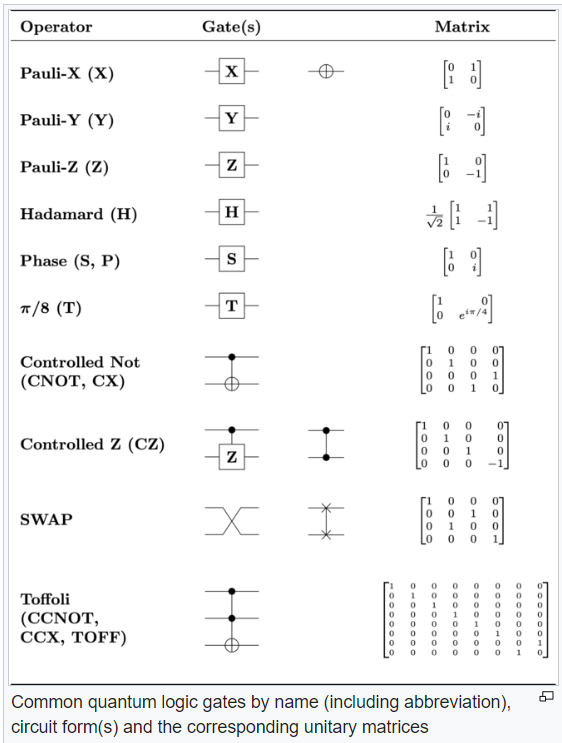
* + - 1. **Phase**

The phase of a quantum state is crucial for interference.  
States with the same magnitude but different phases can interfere differently.

* + 1. **Quantum Gates and Circuits**

In quantum computing, a **quantum logic gate** (or simply **quantum gate**) is a basic quantum circuit operating on a small number of qubits. Quantum logic gates are the building blocks of quantum circuits, like classical logic gates are for conventional digital circuits. Unlike many classical logic gates, quantum logic gates are reversible.

* + - 1. **Single-qubit gates**
* Hadamard gate (H) - Creates superposition
* Pauli gates (X, Y, Z) - Rotations around different axes
* Phase gates (S, T) - Introduce phase shifts  
  + - 1. **Multi-qubit gates**
* Controlled-NOT (CNOT) - Two-qubit gate, fundamental for entanglement
* Toffoli gate - Three-qubit gate, universal for reversible classical computation



*Figure*  ***2***

* + 1. **Quantum Error Correction**

Due to the fragile nature of quantum states, error correction is crucial for reliable quantum computation. Errors comes from Decoherence, gate imperfections and measurement errors. Types of errors are: Bit flips, phase flips, or combinations.

**Correction Techniques**:

* Quantum error correcting codes (e.g., Shor code, Steane code)
* Surface codes
* Topological quantum computing
  + 1. **Quantum Supremacy**   
       It’s the point at which a quantum computer can solve certain computation or optimization problems that is demonstrably beyond the reach of even the most powerful supercomputer. companies are already starting to experiment with quantum computers made by companies like IBM, Rigetti, and D-Wave, a Canadian firm.
    2. **Challenges in Quantum Computing**

Major hurdles in developing practical quantum computers include:

* + - **Scalability**: Building large-scale systems with many qubits
    - Coherence times: Maintaining quantum states long enough for computation
    - **Error rates**: Reducing errors in quantum gates and measurements
    - **Quantum-classical interface**: Efficiently transferring information between systems
    - **Algorithm development:** Creating new algorithms that exploit quantum properties
  1. **Cryptography**

Cryptography is the science and art of securing information by transforming it into a format that is unreadable to unauthorized individuals. It plays a critical role in ensuring the confidentiality, integrity, and authenticity of data, whether it's stored, transmitted, or processed.  
At its core, cryptography involves techniques like encryption and decryption. **Encryption** converts plain, readable data (plaintext) into a scrambled, unreadable format (ciphertext) using a specific algorithm and a key. Only those with the correct key can reverse the process through **decryption** to recover the original data.  
Cryptography has evolved over centuries, from simple ciphers like the Caesar cipher used by Julius Caesar, to complex mathematical algorithms employed in modern computing. Today, cryptography is foundational in securing communication systems, protecting sensitive information, and ensuring secure transactions over the internet.   
Key areas of cryptography include:

* **Symmetric Cryptography**: Both the sender and the receiver use the same key for encryption and decryption. Examples include the Advanced Encryption Standard (AES) and the Data Encryption Standard (DES).
* **Asymmetric Cryptography**: Uses a pair of keys—a public key for encryption and a private key for decryption. This is used in protocols like RSA and ECC (Elliptic Curve Cryptography).
* **Hash Functions**: Cryptographic hash functions generate a fixed-size output (hash) from input data, which is typically used for integrity checks. Common examples include SHA-256 and MD5.
* **Digital Signatures**: Provide a way to verify the authenticity of a message or document, ensuring it was sent by a specific sender and has not been altered.

Cryptography is essential in securing everything from personal communications and financial transactions to national security systems. As technology advances, cryptography continues to adapt, developing new methods to combat emerging threats.

* + 1. **Asymmetric Cryptography**

Public-key cryptography allows communicating parties to establish a shared secret key over a public channel. The concept of public-key cryptography also enables creating digital signatures for authentication of the source and validation of the integrity of data. Asymmetric cryptographical algorithms in use today are: RSA, Elliptic Curve, Diffie Helman, El Gamal All based on some combination of prime factorization or discrete logarithms.

* + 1. **RSA**

RSA (Rivest–Shamir–Adleman) is a public-key cryptosystem, one of the oldest widely used for secure data transmission.   
The initialism "RSA" comes from the surnames of Ron Rivest, Adi Shamir and Leonard Adleman, who publicly described the algorithm in 1977. In a public-key cryptosystem, the **encryption key** is **public** and distinct from the **decryption key**, which is kept secret (**private**). An RSA user creates and publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers are kept secret. Messages can be encrypted by anyone, via the public key, but can only be decrypted by someone who knows the private key.  
The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers, the "factoring problem". Breaking RSA encryption is known as the RSA problem. Whether it is as difficult as the factoring problem is an open question.   
There are no published methods to defeat the system if a large enough key is used. RSA is a relatively slow algorithm. Because of this, it is not commonly used to directly encrypt user data. More often, RSA is used to transmit shared keys for symmetric-key cryptography, which are then used for bulk encryption–decryption.

* + 1. **Elliptic Curve**

Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC allows smaller keys to provide equivalent security, compared to cryptosystems based on modular exponentiation in Galois fields, such as the RSA cryptosystem and ElGamal cryptosystem.

An elliptic curve is a plane curve over a finite field (rather than the real numbers) which consists of the points satisfying the equation:

along with a distinguished point at infinity, denoted ∞. The coordinates here are to be chosen from a fixed finite field of characteristic not equal to 2 or 3, or the curve equation would be somewhat more complicated. This set of points, together with the group operation of elliptic curves, is an abelian group, with the point at infinity as an identity element. The security of elliptic curve cryptography depends on the ability to compute a point multiplication and the inability to compute the multiplicand given the original point and product point. The size of the elliptic curve, measured by the total number of discrete integer pairs satisfying the curve equation, determines the difficulty of the problem.

The primary benefit promised by elliptic curve cryptography over alternatives such as RSA is a smaller key size, reducing storage and transmission requirements. For example, a 256-bit elliptic curve public key should provide comparable security to a 3072-bit RSA public key. Elliptic curves are applicable for key agreement, digital signatures, pseudo-random generators and other tasks. Indirectly, they can be used for encryption by combining the key agreement with a symmetric encryption scheme. They are also used in several integer factorization algorithms that have applications in cryptography, such as Lenstra elliptic-curve factorization.

* + 1. **Diffie Helman**

Diffie–Hellman (DH) key exchange is a mathematical method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as conceived by Ralph Merkle and named after Whitfield Diffie and Martin Hellman.  
Traditionally, secure encrypted communication between two parties required that they first exchange keys by some secure physical means, such as paper key lists transported by a trusted courier. The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric-key cipher.  
Although Diffie–Hellman key exchange itself is a non-authenticated key-agreement protocol, it provides the basis for a variety of authenticated protocols, and is used to provide forward secrecy in **Transport Layer Security's** ephemeral modes (referred to as EDH or DHE depending on the cipher suite).

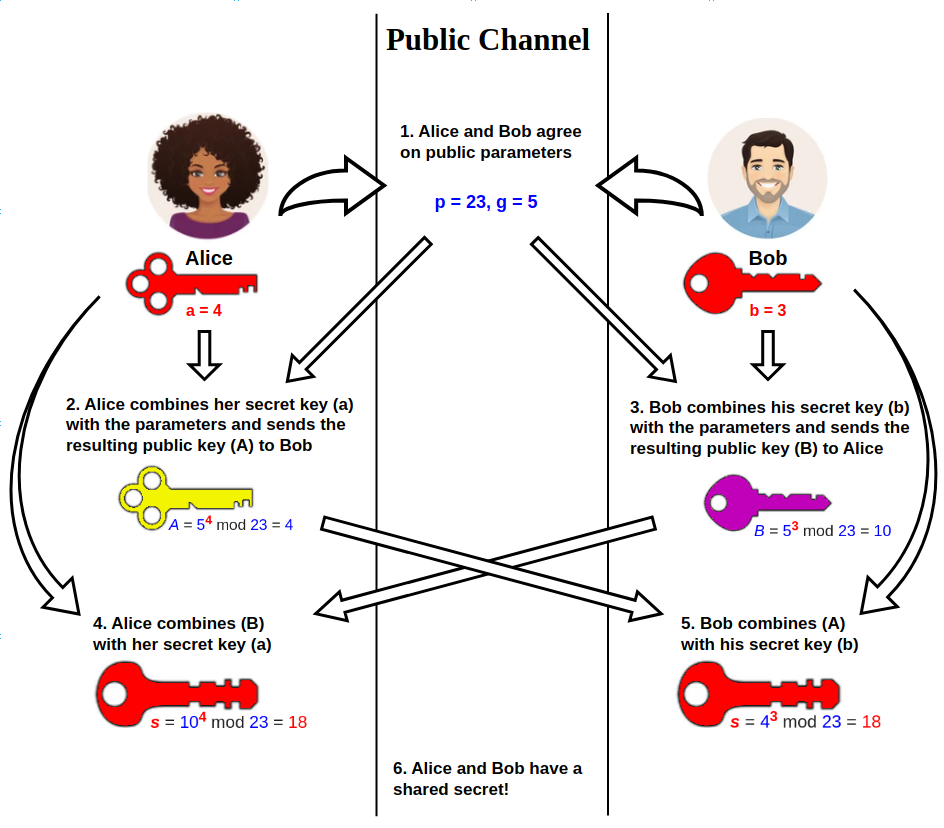


Figure 3

* + 1. **SSL**

SSL (Secure Sockets Layer) and its successor, TLS (Transport Layer Security), are protocols for establishing authenticated and encrypted links between networked computers. Although the SSL protocol was deprecated with the release of TLS 1.0 in 1999, it is still common to refer to these related technologies as “SSL” or “SSL/TLS.” The most current version is TLS 1.3, defined in RFC 8446.

* + 1. **TLS**

Transport Layer Security (TLS) is a cryptographic protocol designed to provide communications security over a computer network. The protocol is widely used in applications such as email, instant messaging, and voice over IP, but its use in securing HTTPS remains the most publicly visible.

The TLS protocol aims primarily to provide security, including privacy (confidentiality), integrity, and authenticity through the use of cryptography, such as the use of certificates, between two or more communicating computer applications. It runs in the presentation layer and is itself composed of two layers: the TLS record and the TLS handshake protocols.

The closely related Datagram Transport Layer Security (DTLS) is a communications protocol that provides security to datagram-based applications. In technical writing, references to "(D)TLS" are often seen when it applies to both versions.

Client-server applications use the TLS protocol to communicate across a network in a way designed to prevent eavesdropping and tampering.

Once the client and server have agreed to use TLS, they negotiate a stateful connection by using a handshaking procedure. The protocols use a handshake with an asymmetric cipher to establish not only cipher settings but also a session-specific shared key with which further communication is encrypted using a symmetric cipher. During this handshake, the client and server agree on various parameters used to establish the connection's security.

* + - 1. **TLS Handshake**

When the connection starts, the record encapsulates a "control" protocol – the handshake messaging protocol. This protocol is used to exchange all the information required by both sides for the exchange of the actual application data by TLS. It defines the format of messages and the order of their exchange. These may vary according to the demands of the client and server – i.e., there are several possible procedures to set up the connection. This initial exchange results in a successful TLS connection (both parties ready to transfer application data with TLS) or an alert message.

1. **Negotiation phase:**

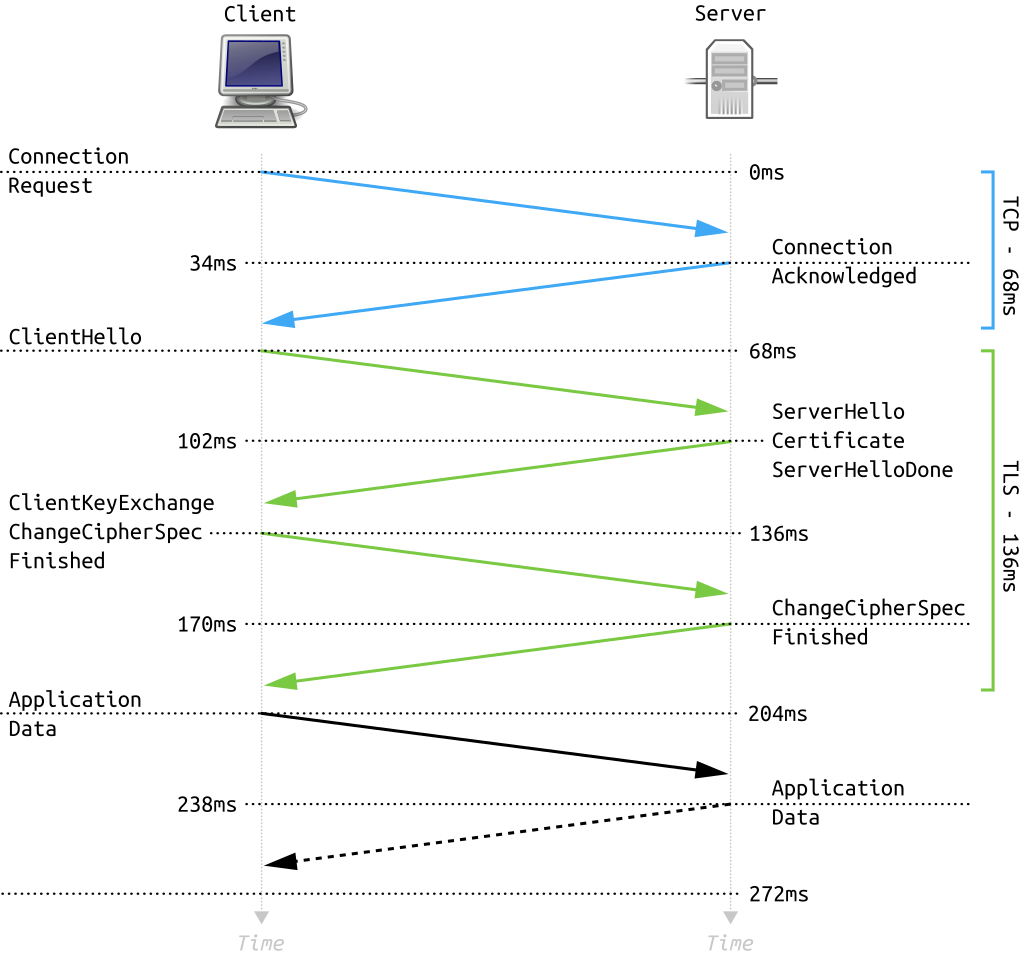
* A client sends a **ClientHello** message specifying the highest TLS protocol version it supports, a random number, a list of suggested cipher suites and suggested compression methods. If the client is attempting to perform a resumed handshake, it may send a session ID. If the client can use Application-Layer Protocol Negotiation, it may include a list of supported application protocols, such as HTTP/2.
* The server responds with a **ServerHello** message, containing the chosen protocol version, a random number, cipher suite and compression method from the choices offered by the client.  
  To confirm or allow resumed handshakes the server may send a session ID. The chosen protocol version should be the highest that both the client and server support.
* The server sends its **Certificate** message (depending on the selected cipher suite, this may be omitted by the server).
* The server sends its **ServerKeyExchange** message (depending on the selected cipher suite, this may be omitted by the server). This message is sent for all DHE, ECDHE and DH\_anon cipher suites.
* The server sends a **ServerHelloDone** message, indicating it is done with handshake negotiation.
* The client responds with a **ClientKeyExchange** message, which may contain a **PreMasterSecret**, public key, or nothing. (Again, this depends on the selected cipher.) This **PreMasterSecret** is encrypted using the public key of the server certificate.
* The client and server then use the random numbers and **PreMasterSecret** to compute a common secret, called the "master secret". All other key data (session keys such as IV, symmetric encryption key, MAC key) for this connection is derived from this master secret (and the client- and server-generated random values), which is passed through a carefully designed pseudorandom function.

1. The client now sends a **ChangeCipherSpec** record, essentially telling the server, "Everything I tell you from now on will be authenticated (and encrypted if encryption parameters were present in the server certificate)." The **ChangeCipherSpec** is itself a record-level protocol with content type of 20.

* The client sends an authenticated and encrypted Finished message, containing a hash and MAC over the previous handshake messages.
* The server will attempt to decrypt the client's Finished message and verify the hash and MAC. If the decryption or verification fails, the handshake is considered to have failed and the connection should be terminated.

1. Finally, the server sends a **ChangeCipherSpec**, telling the client, "Everything I tell you from now on will be authenticated (and encrypted, if encryption was negotiated)."

* The server sends its authenticated and encrypted Finished message.
* The client performs the same decryption and verification procedure as the server did in the previous step.

1. **Application phase**: at this point, the "handshake" is complete and the application protocol is enabled, with content type of 23. Application messages exchanged between client and server will also be authenticated and optionally encrypted exactly like in their Finished message. Otherwise, the content type will return 25 and the client will not authenticate
   1. **Post Quantum Cryptography**Post-quantum cryptography (PQC) refers to cryptographic algorithms believed to be secure against attacks by both classical and quantum computers. While quantum computers excel at certain tasks like factoring large numbers (threatening RSA) and solving discrete logarithm problems (endangering elliptic curve cryptography), they are not superior for all computational tasks. Many complex problems remain challenging even for quantum computers, forming the basis for post-quantum cryptographic systems.   
      Several promising approaches to PQC have emerged:

* **Lattice-based cryptography**: Based on the difficulty of solving certain problems in high-dimensional lattices, such as the shortest vector problem. These systems offer relatively small key sizes and efficient operations.
* **Hash-based cryptography**: Relies on the security of cryptographic hash functions, which are believed to be quantum-resistant. Particularly useful for digital signatures, though they can have large signature sizes.
* **Supersingular isogeny-based cryptography**: Utilizes the complexity of finding isogenies between supersingular elliptic curves. It offers the smallest key sizes among post-quantum systems but can be computationally intensive.
* **Code-based cryptography**: Based on the difficulty of decoding general linear codes. These systems have been studied for decades and offer fast encryption and decryption, but typically have large key sizes.
* **Multivariate cryptography**: Relies on the difficulty of solving systems of multivariate polynomial equations over finite fields. Generally fast for encryption and signatures but often has large key sizes.

Each of these approaches has its strengths and weaknesses in terms of key size, computational efficiency, and security assumptions, and research is ongoing to refine and validate these systems for widespread use in a post-quantum world.

**Stopped here**

**Might be used later:   
Standardization of post-quantum cryptography**

**The worldwide effort for developing and standardizing is centred around the**[**NIST Post-Quantum Cryptography Standardization Project**](https://csrc.nist.gov/projects/post-quantum-cryptography)**. In 2016, the NIST PQC project issued a call for proposals for quantum-resistant digital signature and key encapsulation mechanisms, kicking off a multi-year project to standardize one or more quantum-resistant cryptosystems after several rounds of public review and comment. In 2022, NIST announced its selection of 4 algorithms for standardization: the key encapsulation mechanism CRYSTALS-Kyber, and three signature schemes CRYSTALS-Dilithium, Falcon, and SPHINCS+. In 2023, NIST released draft standards for 3 of those algorithms, with a goal of publishing those standards in 2024. NIST continues to evaluate additional post-quantum algorithms for potential standardization.**

**Standardization of post-quantum algorithms is also taking place in other bodies. The**[**Crypto Forum Research Group**](https://datatracker.ietf.org/rg/cfrg/about/)**within the Internet Engineering Task Force has standardized two stateful hash-based signature schemes (XMSS and LMS/HSS). The International Organization for Standardization (ISO) is also considering the standardization of several post-quantum algorithms.**

**Key Aspects of OQS:**

**Open Source Nature**:

OQS is an **open-source initiative**, meaning that its codebase is publicly accessible and can be reviewed, modified, and contributed to by the community.

The project encourages transparency, peer review, and collaboration.

**Quantum-Resistant Algorithms**:

OQS focuses on researching, implementing, and standardizing cryptographic algorithms that remain secure even in the presence of powerful quantum computers.

These algorithms are designed to withstand attacks from both classical and quantum adversaries.

**Components of OQS**:

**liboqs**: The heart of the project is the **liboqs** library, written in C. It provides a collection of quantum-resistant cryptographic algorithms.

**Prototype Integrations**: OQS integrates its algorithms into widely used cryptographic libraries and protocols. For instance, it has prototype integrations with **OpenSSL**.

**Supported Algorithms**:

OQS includes a variety of post-quantum cryptographic algorithms, such as:

**Lattice-based algorithms**: These rely on the hardness of lattice problems.

**Code-based algorithms**: These use error-correcting codes for security.

**Multivariate-quadratic-equations (MQ)** schemes.

**Isogeny-based algorithms**: Leveraging isogenies between elliptic curves.

**Hash-based algorithms**: Based on hash functions.

**Supersingular Isogeny Diffie-Hellman (SIDH)**: A key exchange protocol.

And more!

**Standardization Efforts**:

Some OQS algorithms are candidates for **NIST’s Post-Quantum Cryptography Standardization** process.

The project actively participates in discussions and submissions to NIST.

**Community Involvement**:

OQS welcomes contributions from researchers, developers, and enthusiasts.

Collaboration occurs through the project’s **GitHub repository**, where issues, pull requests, and discussions take place.

**Conclusion:**

The OQS project plays a crucial role in preparing our cryptographic infrastructure for the quantum era. By fostering collaboration, promoting open-source solutions, and advancing quantum-resistant algorithms, OQS contributes to a more secure digital landscape.

[Quantum computers are still in the experimental stage, but they hold the potential to revolutionize fields like cryptography, optimization, and material science](https://www.technologyreview.com/2019/01/29/66141/what-is-quantum-computing/).

**“Post-quantum Key Exchange for the Internet and the Open Quantum Safe Project”**:

This paper provides insights into post-quantum cryptography, focusing on key exchange protocols.

[It reviews two quantum-resistant key exchange protocols based on lattice problems: **BCNS15** (ring learning with errors problem) and **Frodo** (learning with errors problem)1](https://link.springer.com/content/pdf/10.1007/978-3-319-69453-5_2.pdf).

**Post-quantum Key Exchange for the Internet**:

**Quantum Threat**: Quantum computers could break existing public key cryptosystems, necessitating quantum-resistant cryptography.

[**Lattice-Based Cryptography**: BCNS15 and Frodo are key exchange protocols based on lattice problems (ring-LWE and LWE)1](https://edgeservices.bing.com/edgesvc/chat?udsframed=1&form=SHORUN&clientscopes=chat,noheader,udsedgeshop,channelstable,ntpquery,devtoolsapi,udsinwin11,udsdlpconsent,udscstart,cspgrd,&shellsig=db3f62ced76a8e89800fada9c33d0d34a284184a&setlang=he&lightschemeovr=1&udsps=0&udspp=0#sjevt%7CDiscover.Chat.SydneyClickPageCitation%7Cadpclick%7C0%7C3dbe3080-3f52-476a-aa80-ac55826d6aaa).

**Performance Comparison**:

**BCNS15**: Good tradeoff between performance and key size.

**Frodo**: Fast, but larger communication.

**SIDH**: Small keys, slow performance.

[**Outlook**: Exciting times ahead for designing and standardizing post-quantum cryptography](https://edgeservices.bing.com/edgesvc/chat?udsframed=1&form=SHORUN&clientscopes=chat,noheader,udsedgeshop,channelstable,ntpquery,devtoolsapi,udsinwin11,udsdlpconsent,udscstart,cspgrd,&shellsig=db3f62ced76a8e89800fada9c33d0d34a284184a&setlang=he&lightschemeovr=1&udsps=0&udspp=0#sjevt%7CDiscover.Chat.SydneyClickPageCitation%7Cadpclick%7C1%7C3dbe3080-3f52-476a-aa80-ac55826d6aaa).

**Updates from the Open Quantum Safe Project**:

[Presented at the **NIST Computer Security Conference**, this document discusses the ongoing work of the OQS project in the field of quantum-safe cryptography2](https://csrc.nist.gov/CSRC/media/Events/third-pqc-standardization-conference/documents/accepted-papers/schanck-open-quantum-safe-project-pqc2021.pdf).

Links:

Introduction - [Explainer: What is a quantum computer? | MIT Technology Review](https://www.technologyreview.com/2019/01/29/66141/what-is-quantum-computing/)