Quantum Cryptography

PH7024 Graduate Quantum Mechanics Term Project

Geraldine Chong

April 17, 2023

Abstract

This report explores how quantum mechanics plays a role in the development of more secure cryptography algorithms, a field known as quantum cryptography. The BB84 (developed by Charles Bennett and Gilles Brassard in 1984) and the E91 (proposed by Artur Ekert in 1991) protocols are discussed. The BB84 protocol is then simulated and numerical results of error rates in the cases of (1) no eavesdropper, and (2) an eavesdropper, reported and compared to theoretical findings.

1. Introduction

The use of the Internet has become more prevalent due to rapid advancements in technology and the ease of communication. However, the more reliant the world becomes on the Internet, the more information is transmitted through the networks, which brings about a new risk: Information security. For this reason, the growth of the Internet also compels the improvement of measures used to protect our data. Namely, the use of data encryption.

While classical encryption techniques have worked well, with the recent research progress into building quantum computers and quantum technologies, they are no longer that secure. Information classically encrypted via variants of integer factorization can be decrypted by quantum computers much faster than classical computers through the use of quantum computing algorithms [1].

Therefore, the implementation of quantum cryptography is crucial to ensure future information security. This report explores and investigates how quantum cryptography exploits the laws of quantum mechanics to create secure private keys. It discusses two well-known quantum key distribution algorithms, BB84 and E91, then simulates the BB84 protocol as an in-depth study.

2. Quantum Key Distribution Algorithms

Quantum key distribution (QKD), also known as quantum cryptography, is a provably secure protocol that builds on the principles of quantum mechanics to securely distribute private information [2]. It involves the creation of private key bits, but shared between two parties over a *public* channel.

In QKD algorithms, the bits that make up the private key can be encoded in e.g. the polarization state of a photon and sent over a quantum channel (usually optical fiber). Since the qubit is in a superposition state, any measurement will cause it to collapse into a well-defined state. Additionally, the no-cloning theorem [3] postulates that it is impossible to create an identical copy of an arbitrary state. Therefore, there is no way for an eavesdropper to intercept the qubit without being detected; This is what makes QKD very secure.

Two QKD algorithms, BB84 and E91, are discussed in this section.

2.1. BB84 Protocol

In 1984, Charles Bennett and Gilles Brassard developed the earliest quantum key distribution protocol that was based on Heisenberg's uncertainty principle [4]. They reasoned that when information is encoded in non-orthogonal quantum states, such as the polarization directions of a single photon, then the information cannot be read or copied reliably by an eavesdropper while in transmission.

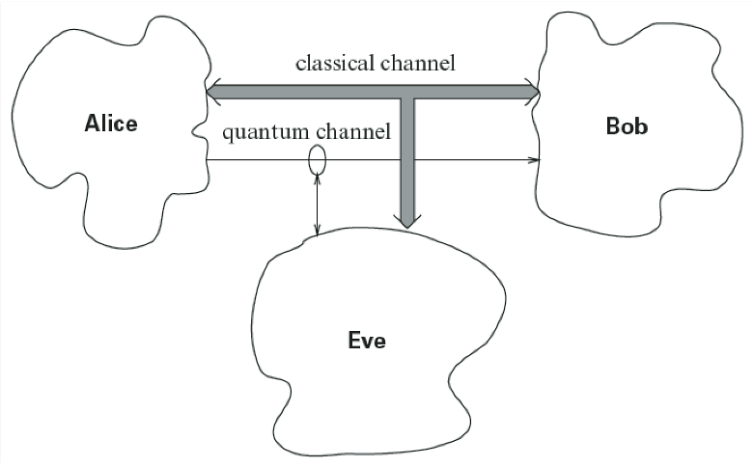

Figure 1: BB84 QKD protocol. Image credited to [5].

Figure 1 shows the situational setup for the BB84 protocol. The aim is to establish a symmetric secret key (of bits 0 and 1) between two parties, Alice and Bob, who will then use it to encrypt and decrypt private correspondences. While the protocol ensures that the key is secure, it does not guarantee success in establishing the key.

To briefly describe, Alice and Bob are connected by two public channels: a classical channel and a quantum channel. Alice first generates a sequence of $(4+d)n$ random classical bits (0 and 1). She then encodes each bit in the sequence in a polarization state of a photon, chosen randomly. The two bases used for encoding are:

$$\begin{array}{cc} \textbf{Computational (Z) Basis:} & \textbf{Hadamard (X) Basis:} \\ H \longrightarrow |{\rightarrow}\rangle & DL \longrightarrow |{\nwarrow}\rangle = \dfrac{1}{\sqrt{2}}\left(|H\rangle - |V\rangle\right) \\ V \longrightarrow |{\uparrow}\rangle & DR \longrightarrow |{\nearrow}\rangle = \dfrac{1}{\sqrt{2}}\left(|H\rangle + |V\rangle\right) \end{array} \tag{1}$$

Alice sends this sequence of photons to Bob through the quantum channel. They confirm through the classical channel that Bob has received all the photons sent by Alice. Bob then measures ("decodes") the state of each photon in the sequence by picking a random basis (computational or Hadamard).

Once done, Alice and Bob share with each other their choice of encoding and decoding bases. For the choice of bases where both chose the same, Bob's decoded bit will match Alice's initial bit. For when the choice of bases disagree, there is only a 50% chance that Bob's bit matches Alice's initial bit. Thus, they discard the bits where the choice of bases disagree.

Alice and Bob then pick from the leftover bits a subset of n bits and announce them to check if the bit values match, and if any eavesdropping occurred. The checked bits are discarded, and the remaining bits form their secret key. An example realization of this procedure is illustrated in Figure 2.

QUANTUM TRANSMISSION															
Alice's random bits	0	1	1	0	1	1	0	0	1	0	1	1	0	0	1
Random sending bases	D	R	D	R	R	R	R	R	D	D	R	D	D	D	R
Photons Alice sends	↗	↕	↘	↔	↕	↕	↔	↔	↘	↗	↕	↘	↗	↗	↕
Random receiving bases	R	D	D	R	R	D	D	R	D	R	D	D	D	D	R
Bits as received by Bob	1		1		1	0	0	0		1	1	1		0	1
PUBLIC DISCUSSION															
Bob reports bases of received bits	R		D		R	D	D	R		R	D	D		D	R
Alice says which bases were correct			OK		OK			OK				OK		OK	OK
Presumably shared information (if no eavesdrop)			1		1			0				1		0	1
Bob reveals some key bits at random					1									0	
Alice confirms them					OK									OK	
OUTCOME															
Remaining shared secret bits			1					0				1			1

Figure 2: Sample realization of the BB84 protocol. Image credited to [4].

Since Alice and Bob discard all the bits in which their choice of bases disagree, then if there is no eavesdropper, the theoretical error rate is 0 — All checked bits should match.

However, if there has been an eavesdropper, Eve, present, then there will be mismatches in some of the checked bits. When Eve intercepts Alice's photon, she has a 50% chance of picking the same basis (just like if Bob were to measure it first). Choosing the wrong basis would mean that Eve has disturbed the initial state of the photon sent by Alice, before sending it on to Bob. In this situation, no matter which basis Bob measures the photon in, he will only have a 50% chance of obtaining the correct bit value. The theoretical error rate (in the event of eavesdropping) is thus computed:

$$\begin{aligned} E_{tot} &= E_{eve} \times E_{bob} \\ &= \frac{1}{2} \times \frac{1}{2} = \frac{1}{4} \\ &= 0.25 \end{aligned} \tag{2}$$

Therefore, if Alice and Bob check their subset of bits and find that the error rate is ~ 0.25, they will know that Eve was present. They will then abandon the current key and repeat the protocol to establish a new key.

2.2. E91 Protocol

Later, in 1991, Artur Ekert proposed a variant of QKD that employs the properties of quantum entanglement, based on the EPR thought experiment [6]. In a nutshell, two entangled particles are considered as one whole, non-separable system, where the state of one particle cannot be described without the other. The entanglement remains intact until a measurement is made on the entangled state. Furthermore, the measurement results of the particles are also correlated.

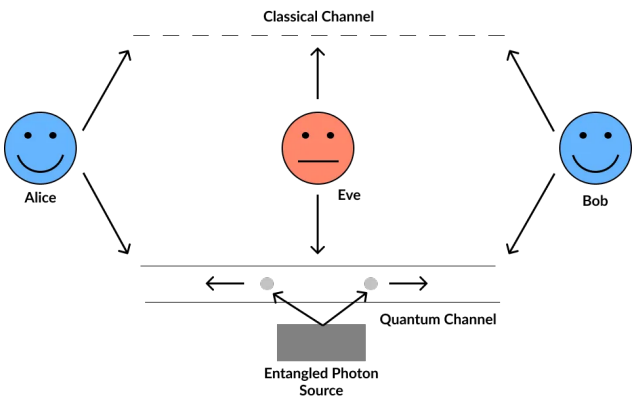

Figure 3: E91 QKD protocol. Image credited to [7].

Figure 3 shows the situational setup for the E91 protocol. Rather than generating a random sequence of classical bits, a source centre prepares a sequence of entangled e.g. $|\Phi^+\rangle$ states [8]:

$$|\Phi^+\rangle = \frac{(|0_A 0_B\rangle + |1_A 1_B\rangle)}{\sqrt{2}} \tag{3}$$

It is clear that $|\Phi^+\rangle$ is a non-separable state (it cannot be written as a tensor product of two states). It is in a superposition of $|0_A 0_A\rangle$ and $|1_A 1_A\rangle$, thus both particles are either in the 0 state or the 1 state. However, it is impossible to know which before a measurement is made. The source then sends the first particle (A) of each state to Alice and the second particle (B) to Bob.

$$a = \left\{0, \frac{\pi}{4}, \frac{\pi}{2}\right\} \qquad\qquad b = \left\{\frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\} \tag{4}$$

Alice then measures her particle in a random basis from a and Bob measures his particle in a random basis from b, where the basis vectors lie in the $x-y$ plane and are characterized by azimuthal angles of a_i or b_j from the computational basis. They record the results from their measurements and share their choice of measurement bases with each other [9]. Based on Equation 4, there is a $\frac{1}{3}$ chance that they chose the same measurement basis each time.

The entangled state collapses into either $|0_A 0_A\rangle$ and $|1_A 1_A\rangle$ upon measurement. For instances where their choice of measurement bases agree, then when Alice measures 0, Bob will also measure 0 with 100% probability (and vice versa for 1).

Alice and Bob separate their results into two groups: (1) Key qubits, where they chose the same measurement bases, (2) Decoy qubits, where their choice of measurement bases disagree.

Using the second group of results, the correlation coefficient (Equation 5) is found to be:

$$E(a_i, b_j) = P_{++}(a_i, b_j) + P_{--}(a_i, b_j) - P_{+-}(a_i, b_j) - P_{-+}(a_i, b_j) \tag{5}$$

The quantity S (sum of all correlation coefficients where bases disagree) can then be computed:

$$S = E(a_1, b_1) - E(a_1, b_3) + E(a_3, b_1) + E(a_3, b_3) \tag{6}$$

Quantum mechanics requires that $S = 2\sqrt{2}$. Thus, if there was no eavesdropper present, then the quantity should be as expected. However, if there was an eavesdropper present, then the quantity should not violate Bell's inequality e.g. $S \leq 2$.

3. Simulating the BB84 Protocol

To better visualize and understand how BB84 works, we simulate the protocol for two cases: (1) without an eavesdropper, and (2) with an eavesdropper. The error rates for both cases are also calculated and compared to their respective theoretical predictions.

3.1. Overview of the Simulation

Our simulation references an experimental setup where the information Alice sends is encoded in the polarization state of the photon [5]. A typical experimental setup is referenced in Figure 4.

Our simulation largely follows the same setup, but the notations differ in which we use the encoding bases defined earlier in Equation 1.

The setup includes a mirror (which reflects or transmits the photon depending on the basis that Bob mea-

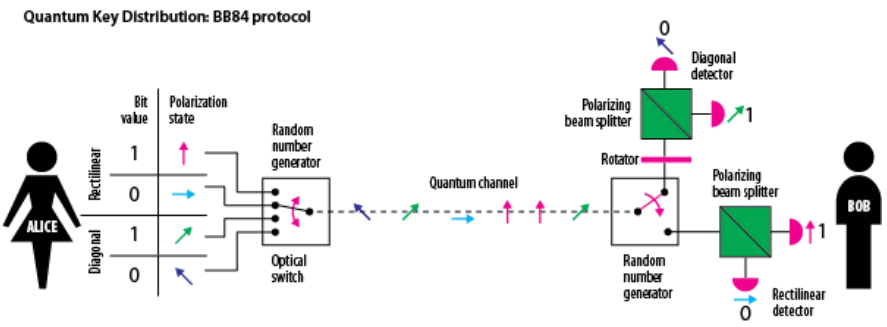

Figure 4: A typical BB84 experiment setup. Image credited to [10].

sures the photon in), a rotator (which rotates the state of the photon in the case of measurement using the Hadamard basis), and a polarizing beam-splitter (PBS, which transmits horizontally polarized photons and reflects vertically polarized photons).

The protocol begins with Alice sending over an encoded bit (either in the computational or the Hadamard basis) to Bob. There are two possibilities from here.

- **Bob measures the photon in the computational basis:** In this case, the photon will not be reflected by the mirror, neither will it encounter the rotator. It will simply go to the polarizing beam-splitter, and depending on its polarization, be transmitted or reflected into its corresponding detector. The possible final states are:

$$
\begin{aligned}
PBS\,|0,H\rangle &= |0,H\rangle \\
PBS\,|1,H\rangle &= |1,H\rangle \\
PBS\,|0,V\rangle &= |0,V\rangle \\
PBS\,|1,V\rangle &= |1,V\rangle
\end{aligned}
\qquad
\begin{aligned}
PBS\,|0,DL\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle - |1,V\rangle\right) \\
PBS\,|1,DL\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle - |0,V\rangle\right) \\
PBS\,|0,DR\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle + |1,V\rangle\right) \\
PBS\,|1,DR\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle + |0,V\rangle\right)
\end{aligned}
\tag{7}
$$

- **Bob measures the photon in the Hadamard basis:** In this case, the photon will first be reflected by the mirror (M), then its polarization will be rotated by the rotator (R). After, it will go to the polarizing beam-splitter (PBS), and depending on its polarization, be transmitted or reflected into its corresponding detector. The possible final states are:

$$
\begin{aligned}
PBS\,(M\otimes R)\,|0,H\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle + |0,V\rangle\right) \\
PBS\,(M\otimes R)\,|1,H\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle + |1,V\rangle\right) \\
PBS\,(M\otimes R)\,|0,V\rangle &= \frac{1}{\sqrt{2}}\left(|0,V\rangle - |1,H\rangle\right) \\
PBS\,(M\otimes R)\,|1,V\rangle &= \frac{1}{\sqrt{2}}\left(|1,V\rangle - |0,H\rangle\right)
\end{aligned}
\qquad
\begin{aligned}
PBS\,(M\otimes R)\,|0,DL\rangle &= |1,H\rangle \\
PBS\,(M\otimes R)\,|1,DL\rangle &= |0,H\rangle \\
PBS\,(M\otimes R)\,|0,DR\rangle &= |0,V\rangle \\
PBS\,(M\otimes R)\,|1,DR\rangle &= |1,V\rangle
\end{aligned}
\tag{8}
$$

The complete derivations of Equations 7 and 8 are included in Appendix A and the supplementary Jupyter notebook [11], but the main conclusion from the above results is that if the photon is measured in the basis it was encoded in, then the final state is in the same polarization. However, if the photon is measured in a different basis from what it was encoded in, then there is an equal chance of detecting it in either polarization.

3.2. Implementing the Simulation

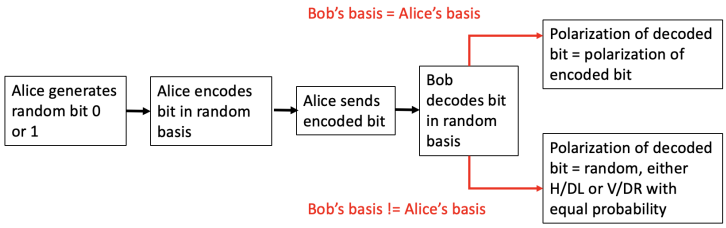

Figure 5: Flowchart of simulation.

The protocol will be simulated based on what we know from the results computed in the previous section. The flowchart in Figure 5 visualizes the implementation step-by-step.

First, an array of **(4+d)n** random bits (0 and 1) is generated. Then, an array of random encoding bases (0 for computational Z basis, 1 for Hadamard X basis) of the same length is generated. Each random bit is encoded in its corresponding basis with the function `encode`:

$$\begin{array}{ll} \textbf{Computational (Z) Basis:} & \textbf{Hadamard (X) Basis:} \\ 0 \longrightarrow H & 0 \longrightarrow DL \\ 1 \longrightarrow V & 1 \longrightarrow DR \end{array} \tag{9}$$

Next, the encoded array of bits is passed into the function `decode`, which generates a random decoding basis (0 for computational Z basis, 1 for Hadamard X basis) for each encoded bit. There are two cases:

- **Computational decoding basis:** If the encoded bit is H or V, then the same result is returned as the decoded bit. If the encoded bit is DL or DR, then it randomly returns H or V as the decoded bit with equal probability.
- **Hadamard decoding basis:** If the encoded bit is DL or DR, then the same result is returned as the decoded bit. If the encoded bit is H or V, then it randomly returns DL or DR as the decoded bit with equal probability.

The function `decode` returns both the array of decoded bits and the array of decoding bases.

For error checking, the array of encoding bases is checked against the array of decoding bases. Both arrays are passed into a function `match`, which returns an array of indexes whereby the encoding and decoding bases match, the match count, an array of encoded bits at the matching indexes, and an array of decoded bits at the matching indexes.

Finally, the function `compare` takes in the arrays of encoded and decoded bits at matching indexes, and a number **n**. It checks **n** number of bits at random indexes within both arrays and deletes the checked bits. It returns the final arrays of decoded and encoded bits, and the error rate.

The above simulates the case without an eavesdropper. In the case of an eavesdropper, Eve, no additional functions are required, but the function `decode` will be called twice:

- Alice generates bits.
- Alice encodes bits with `encode`.
- Eve decodes bits with `decode`.
- Bob "decodes" bits from Eve with `decode`.
- Alice and Bob perform error checking with `match` and `compare`.

4. Results and Discussion

The complete simulations (preliminary and full) can be found in the supplementary Jupyter notebook [11], and the source code for the functions defined in the implementation is included in Appendix B.

4.1. Preliminary Tests

First, we run mini-tests for both cases to ensure the functions are working. The following values are used as parameters: `n` = 8, `d` = 0. Figure 6 has no eavesdropper. Figure 7 has an eavesdropper.

```
----- SIMULATION START (WITHOUT EAVESDROPPER) -----

ALICE GENERATING 32 BITS......
alice bits: [1 0 1 1 1 0 0 1 1 1 0 1 1 1 0 0 0 1 0 0 1 0 1 1 0 1 0 1 1 0 1 1]

ALICE ENCODING BITS IN RANDOM BASIS......
encoded bits: ['dr', 'h', 'v', 'v', 'dr', 'dl', 'h', 'v', 'dr', 'dr', 'dl', 'dr', 'dr', 'v', 'dl', 'h', 'h', 'dr',
'h', 'h', 'v', 'h', 'dr', 'dr', 'h', 'dr', 'dl', 'dr', 'dr', 'h', 'v', 'v']

BOB DECODING BITS IN RANDOM BASIS......
decoded bits: ['dr', 'h', 'dr', 'dl', 'dr', 'h', 'dl', 'v', 'dr', 'dr', 'h', 'dr', 'dr', 'v', 'h', 'h', 'h', 'dr',
'h', 'h', 'dl', 'h', 'h', 'v', 'h', 'dr', 'v', 'h', 'dr', 'h', 'v', 'v']

ALICE AND BOB ANNOUNCING ENCODING / DECODING BASIS......
encoding basis: [1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0]
decoding basis: [1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0]

ALICE AND BOB KEEP BITS IN WHICH SAME BASIS WAS USED......
basis matched 21/32 times.
indexes where basis match: [0, 1, 4, 7, 8, 9, 11, 12, 13, 15, 16, 17, 18, 19, 21, 24, 25, 28, 29, 30, 31]
matched bits (encoded, 21 bits) = ['dr', 'h', 'dr', 'v', 'dr', 'dr', 'dr', 'dr', 'v', 'h', 'h', 'dr', 'h', 'h', 'h',
'h', 'dr', 'dr', 'h', 'v', 'v']
matched bits (decoded, 21 bits) = ['dr', 'h', 'dr', 'v', 'dr', 'dr', 'dr', 'dr', 'v', 'h', 'h', 'dr', 'h', 'h', 'h',
'h', 'dr', 'dr', 'h', 'v', 'v']

ALICE AND BOB COMPARE A SUBSET OF 8 BITS TO CHECK ERROR RATE......
checked bits (encoded, 8 bits) = ['v', 'dr', 'h', 'h', 'h', 'dr', 'dr', 'h']
checked bits (decoded, 8 bits) = ['v', 'dr', 'h', 'h', 'h', 'dr', 'dr', 'h']

kept bits (encoded, 13 bits) = ['dr', 'dr', 'v', 'dr', 'dr', 'v', 'h', 'dr', 'h', 'h', 'dr', 'h', 'v']
kept bits (decoded, 13 bits) = ['dr', 'dr', 'v', 'dr', 'dr', 'v', 'h', 'dr', 'h', 'h', 'dr', 'h', 'v']
error rate = 0.0
no eavesdropper detected!
```

Figure 6: Preliminary test with no eavesdropper.

```
----- SIMULATION START (WITH EAVESDROPPER) -----

ALICE GENERATING 32 BITS......
alice bits: [0 1 0 0 0 1 1 1 0 0 0 0 1 1 1 1 1 0 1 0 1 1 0 0 0 1 0 0 1 1 0 0]

ALICE ENCODING BITS IN RANDOM BASIS......
encoded bits: ['dl', 'dr', 'dl', 'h', 'h', 'dr', 'v', 'v', 'dl', 'h', 'dl', 'dl', 'v', 'v', 'dr', 'dr', 'v', 'dl', 'd
r', 'h', 'v', 'dr', 'dl', 'dl', 'h', 'v', 'h', 'h', 'dr', 'v', 'dl', 'h']

EVE DECODING BITS IN RANDOM BASIS......
decoded bits BY EVE: ['dl', 'dr', 'dl', 'h', 'h', 'v', 'v', 'dr', 'h', 'h', 'v', 'h', 'dl', 'v', 'h', 'dr', 'v', 'h',
'h', 'h', 'v', 'v', 'dl', 'dl', 'h', 'v', 'dr', 'h', 'h', 'dl', 'dl', 'h']

BOB MEASURING BITS FROM EVE IN RANDOM BASIS......
decoded bits: ['dl', 'dr', 'dl', 'h', 'dl', 'dl', 'dl', 'dr', 'h', 'h', 'v', 'dl', 'dl', 'dr', 'h', 'h', 'v', 'dr',
'dr', 'h', 'v', 'dr', 'dl', 'dl', 'dr', 'dr', 'dr', 'dr', 'h', 'h', 'v', 'h']

ALICE AND BOB ANNOUNCING ENCODING / DECODING BASIS......
encoding basis: [1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0]
eve's dc basis: [1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0]
decoding basis: [1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]

ALICE AND BOB KEEP BITS IN WHICH SAME BASIS WAS USED......
basis matched 17/32 times.
indexes where basis match: [0, 1, 2, 3, 5, 9, 11, 16, 17, 18, 19, 20, 21, 22, 23, 29, 31]
matched bits (encoded, 17 bits) = ['dl', 'dr', 'dl', 'h', 'dr', 'h', 'dl', 'v', 'dl', 'dr', 'h', 'v', 'dr', 'dl', 'd
l', 'v', 'h']
matched bits (decoded, 17 bits) = ['dl', 'dr', 'dl', 'h', 'dl', 'h', 'dl', 'v', 'dr', 'dr', 'h', 'v', 'dr', 'dl', 'd
l', 'h', 'h']

ALICE AND BOB COMPARE A SUBSET OF 8 BITS TO CHECK ERROR RATE......
checked bits (encoded, 8 bits) = ['h', 'v', 'dr', 'v', 'dr', 'h', 'dr', 'dl']
checked bits (decoded, 8 bits) = ['h', 'v', 'dr', 'v', 'dl', 'h', 'dr', 'dl']

kept bits (encoded, 9 bits) = ['dl', 'h', 'dl', 'dl', 'h', 'dr', 'dl', 'dl', 'v']
kept bits (decoded, 9 bits) = ['dl', 'h', 'dl', 'dr', 'h', 'dr', 'dl', 'dl', 'h']
error rate = 0.125
eavesdropper detected!
```

Figure 7: Preliminary test with an eavesdropper.

4.2. Full Simulation

The following values are used as parameters: `n` = 1000, `d` = 6. Figure 8 has no eavesdropper. Figure 9 has an eavesdropper.

```
----- SIMULATION START (WITHOUT EAVESDROPPER) -----

ALICE GENERATING 10000 BITS......

ALICE ENCODING BITS IN RANDOM BASIS......

BOB DECODING BITS IN RANDOM BASIS......

ALICE AND BOB ANNOUNCING ENCODING / DECODING BASIS......

ALICE AND BOB KEEP BITS IN WHICH SAME BASIS WAS USED......
basis matched 5053/10000 times.

ALICE AND BOB COMPARE A SUBSET OF 1000 BITS TO CHECK ERROR RATE......
error rate = 0.0
no eavesdropper detected!
```

Figure 8: Full simulation with no eavesdropper.

```
----- SIMULATION START (WITH EAVESDROPPER) -----

ALICE GENERATING 10000 BITS......

ALICE ENCODING BITS IN RANDOM BASIS......

EVE DECODING BITS IN RANDOM BASIS......

BOB MEASURING BITS FROM EVE IN RANDOM BASIS......

ALICE AND BOB ANNOUNCING ENCODING / DECODING BASIS......

ALICE AND BOB KEEP BITS IN WHICH SAME BASIS WAS USED......
basis matched 5035/10000 times.

ALICE AND BOB COMPARE A SUBSET OF 1000 BITS TO CHECK ERROR RATE......
error rate = 0.258
eavesdropper detected!
```

Figure 9: Full simulation with an eavesdropper.

4.3. Discussion

It can be seen that in the case without an eavesdropper, the error rate is 0 (assuming a lossless quantum channel, and Bob receives every bit from Alice). This agrees with the theoretical prediction. The bits in which the encoding and decoding basis do not match have been discarded, therefore, all the remaining bits should match and the error rate should be 0.

In the case of an eavesdropper, the error rate is 0.258. This result also agrees with theoretical predictions; Eve has a 50% chance of measuring the bit in Alice's encoding basis, and Bob has a 50% chance of obtaining the correct bit no matter which basis he uses. The error rate should thus be around $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4} = 0.25$.

5. Conclusion

The two protocols mentioned in this report are fundamental and well-known in quantum cryptography, but newer and more modern protocols have been established since then, that involve quantum entanglement. An example would be the BBM92 protocol [12], proposed by Charles Bennett, Gilles Brassard and David Mermin in 1992, which makes use of polarized entangled photon pairs in the generation of the secret key.

As discussed, the properties of quantum entanglement are crucial in advancing QKD, due to information encoded in entangled states being inherently private [13] and thus secure. Research on how to improve and develop better quantum cryptography algorithms is still ongoing, and there is much yet to be discovered.

References

[1] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Journal on Computing* 26 (5 Oct. 1997). ISSN: 0097-5397. DOI: 10.1137/S0097539795293172.

[2] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.

[3] W. K. Wootters and W. H. Zurek. "A single quantum cannot be cloned". In: 299.5886 (Oct. 1982), pp. 802–803. DOI: 10.1038/299802a0.

[4] Charles H. Bennett and Gilles Brassard. "Quantum cryptography: Public key distribution and coin tossing". In: *Theoretical Computer Science* 560 (2014). Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84, pp. 7–11. ISSN: 0304-3975. DOI: https://doi.org/10.1016/j.tcs.2014.05.025. URL: https://www.sciencedirect.com/science/article/pii/S0304397514004241.

[5] Eleanor Rieffel and Wolfgang Polak. *Quantum Computing – A Gentle Introduction*. Mar. 2011. ISBN: 978-0-262-01506-6.

[6] Artur K. Ekert. "Quantum cryptography based on Bell's theorem". In: *Physical Review Letters* 67.6 (Aug. 1991), pp. 661–663. DOI: 10.1103/physrevlett.67.661. URL: https://doi.org/10.1103/physrevlett.67.661.

[7] Maanav Seth and Amit Yadav. *Fundamentals of Quantum Key Distribution — BB84, B92 E91 protocols*. 2021. URL: https://medium.com/@qcgiitr/fundamentals-of-quantum-key-distribution-bb84-b92-e91-protocols-e1373b683ead.

[8] Ravi Kashyap. *Quantum Key Distribution: Ekert E91 Protocol Derivations*. Dec. 2022. DOI: http://dx.doi.org/10.2139/ssrn.4299689.

[9] N Ilic. "The Ekert Protocol". In: *Journal of PHY334* (334 2007). URL: https://www.ux1.eiu.edu/~nilic/Nina's-article.pdf.

[10] Department of Physics UNS Nice (France). *Protocole BB84*. 2015.

[11] Geraldine Chong. *Supplementary Jupyter Notebook*. 2023. URL: https://github.com/gravitance/school/tree/main/coursework.

[12] Charles H. Bennett, Gilles Brassard, and N. David Mermin. "Quantum cryptography without Bell's theorem". In: 68.5 (Feb. 1992), pp. 557–559. DOI: 10.1103/PhysRevLett.68.557.

[13] Stephanie Wehner, David Elkouss, and Ronald Hanson. "Quantum internet: A vision for the road ahead". In: *Science* 362 (6412 Oct. 2018). ISSN: 0036-8075. DOI: 10.1126/science.aam9288.

Appendix

A. Overview of Protocol

The states are defined as follows:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{10}$$

The encoding bases are defined as follows:

$$|H\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad |V\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad |DL\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \qquad |DR\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{11}$$

We find all the possible encoded states by:

$$\begin{aligned} |0,H\rangle = |0\rangle \otimes |H\rangle &= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} & |0,V\rangle = |0\rangle \otimes |V\rangle &= \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\ |1,H\rangle = |1\rangle \otimes |H\rangle &= \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} & |1,V\rangle = |1\rangle \otimes |V\rangle &= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \\ |0,DL\rangle = |0\rangle \otimes |DL\rangle &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix} & |0,DR\rangle = |0\rangle \otimes |DR\rangle &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\ |1,DL\rangle = |1\rangle \otimes |DL\rangle &= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} & |1,DR\rangle = |1\rangle \otimes |DR\rangle &= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \end{aligned} \tag{12}$$

The matrix operators of the experimental components are:

Mirror:	**Rotator:**	**PBS:**
$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

(13)

A.1. Measurement in the Computational Basis

For ease of writing, we define the initial encoded state as $|\psi_i\rangle$. For measurement in the computational basis, the final state is simply the dot product of the PBS and the initial state:

$$|\psi_f\rangle = PBS\,|\psi_i\rangle \tag{14}$$

Therefore, all possible final states are computed as:

$$\begin{aligned} PBS\,|0,H\rangle &= |0,H\rangle & PBS\,|0,DL\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle - |1,V\rangle\right) \\ PBS\,|1,H\rangle &= |1,H\rangle & PBS\,|1,DL\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle - |0,V\rangle\right) \\ PBS\,|0,V\rangle &= |0,V\rangle & PBS\,|0,DR\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle + |1,V\rangle\right) \\ PBS\,|1,V\rangle &= |1,V\rangle & PBS\,|1,DR\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle + |0,V\rangle\right) \end{aligned} \tag{15}$$

A.2. Measurement in the Hadamard Basis

For measurement in the Hadamard basis, the operator is a dot product of the PBS and the tensor product of the mirror (M) and rotator (R):

$$|\psi_f\rangle = PBS\,(M \otimes R)\,|\psi_i\rangle \tag{16}$$

Therefore, all possible final states are computed as:

$$\begin{aligned} PBS\,(M\otimes R)\,|0,H\rangle &= \frac{1}{\sqrt{2}}\left(|1,H\rangle + |0,V\rangle\right) & PBS\,(M\otimes R)\,|0,DL\rangle &= |1,H\rangle \\ PBS\,(M\otimes R)\,|1,H\rangle &= \frac{1}{\sqrt{2}}\left(|0,H\rangle + |1,V\rangle\right) & PBS\,(M\otimes R)\,|1,DL\rangle &= |0,H\rangle \\ PBS\,(M\otimes R)\,|0,V\rangle &= \frac{1}{\sqrt{2}}\left(|0,V\rangle - |1,H\rangle\right) & PBS\,(M\otimes R)\,|0,DR\rangle &= |0,V\rangle \\ PBS\,(M\otimes R)\,|1,V\rangle &= \frac{1}{\sqrt{2}}\left(|1,V\rangle - |0,H\rangle\right) & PBS\,(M\otimes R)\,|1,DR\rangle &= |1,V\rangle \end{aligned} \tag{17}$$

The complete workings deriving Equations 15 and 17 can be found in the supplementary Jupyter notebook [11], but the main conclusion from the above results is that if the photon is measured in the basis it was encoded in, then the final state is in the same polarization. However, if the photon is measured in a different basis from what it was encoded in, then there is an equal chance of detecting it in either polarization.

B. Functions Defined

```
def encode(bitlist):

    encoded = []
    ebase = []

    for bit in bitlist:

        base = int(np.random.choice([0,1]))
        ebase.append(base)

        if base == 0:   # computational basis +
            if bit == 0:   # horizontal
                ebit = "h"
            else:          # vertical
                ebit = "v"

        else:           # hadamard basis X
            if bit == 0:   # diagonal left
                ebit = "dl"
            else:          # diagonal right
                ebit = "dr"

        encoded.append(ebit)

    return encoded, ebase
```

Figure 10: `encode` function

```
def decode(bitlist):

    decoded = []
    dbase = []

    for bit in bitlist:

        base = int(np.random.choice([0,1]))
        dbase.append(base)

        if base == 0:   # computational basis +
            if bit == "h" or bit == "v":   # encode basis = decode basis
                dbit = bit
            else:                          # encode basis != decode basis

                # randomly decide on v or h in computational basis
                randnum = int(np.random.choice([0,1]))
                if randnum == 0:
                    dbit = "h"
                else:
                    dbit = "v"

        else:           # hadamard basis X
            if bit == "dl" or bit == "dr":   # encode basis = decode basis
                dbit = bit
            else:                            # encode basis != decode basis

                # randomly decide on dl or dr in hadamard basis
                randnum = int(np.random.choice([0,1]))
                if randnum == 0:
                    dbit = "dl"
                else:
                    dbit = "dr"

        decoded.append(dbit)

    return decoded, dbase
```

Figure 11: `decode` function

```
def match(ebase, dbase):

    matchindex = []
    matchcount = 0

    for i in range(len(ebase)):
        if ebase[i] == dbase[i]:
            matchindex.append(i)
            matchcount += 1

    ematch = []
    dmatch = []

    for index in matchindex:
        ematch.append(encoded[index])
        dmatch.append(decoded[index])

    return matchindex, matchcount, ematch, dmatch
```

Figure 12: `match` function

```
def compare(ematch, dmatch, n):

    efinal = ematch
    dfinal = dmatch

    totalindex = list(range(len(ematch)))
    random.shuffle(totalindex)

    subsetindex = totalindex[:n]
    subsetindex.sort()
    subsetindex.reverse()

    esub = []
    dsub = []
    count = 0

    for i in subsetindex:
        esub.append(ematch.pop(i))
        dsub.append(dmatch.pop(i))

    for j in range(len(subsetindex)):
        if esub[j] != dsub[j]:
                count += 1

    error = count/len(subsetindex)

    detected = "no eavesdropper detected!"

    if abs((0.25 - error)/0.25) ¡ 0.9:
        detected = "eavesdropper detected!"

    return esub, dsub, efinal, dfinal, error, detected
```

Figure 13: `compare` function