Quantum Computing: Eavesdropping Detection of Encrypted Messages Over Optical Channels With Spatiotemporal Information Loss

Jai Veilleux ◆ Department of Physics ◆ Edmonds College ◆ WA USA

Abstract

As a computer science major the problem of modern encryption systems being vulnerable to attack by quantum computers is extremely interesting. The math underlying the algorithms can be "solved" by a mature quantum computer using Shor's algorithm in logarithmic time, as opposed to the exponential time required by classical computers. By harnessing the power of quantum information systems, we can guarantee that no observer is intercepting our communications due to the no-cloning property of quantum states and mitigate the risk of future decryption. In the event of an observer, significantly more errors will occur during key transmission ensuring detection. At what point does this become indistinguishable from information loss due to distance of signal transmission? I was able to determine that a 5mW laser was capable of transmitting a stable signal up to at least 3m with a 100% laser detection rate and 56% valid base pairs rate.

Introduction

Quantum states can be mapped to classical states using polarized light and Bra-Ket (Dirac) notation. We can use this to represent binary bits from polarized light.

$|v\rangle$ = columnal vector representing a quantum state (Ket)

$\langle f|$ = linear map which maps a vector to a number in the complex plane, row vector (Bra)

$\langle f|v\rangle$ = scalar product of two states ($\langle Bra|Ket\rangle$)

A scalar product of the same states: $\langle 90°|90°\rangle = 1$

While orthogonal states: $\langle 90°|0°\rangle = 0$

From this, we can represent states as combinations of other bases: $|45°\rangle = \frac{1}{\sqrt{2}}|0°\rangle + \frac{1}{\sqrt{2}}|90°\rangle$

Why $\frac{1}{\sqrt{2}}$? $\langle 45°|45°\rangle = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}}\end{bmatrix} = 1$

Squaring the absolute value of a scalar product gives the probability of the expected result from a Ket in decimal form. For the purposes of this experiment, that the received bit matches the transmitted bit. The 0 from orthogonal states is used to detect the 0-bit.

$$|\langle 45°|0°\rangle|^2 = \left|\frac{1}{\sqrt{2}}\langle 45°|45°\rangle + \frac{1}{\sqrt{2}}\langle 45°|-45°\rangle\right|^2 = 0.5$$

This has important consequences, when the incident of polarized light does not match there is only a 50% chance that the data received is correct. Introducing a third-party observer which must randomly select a base decreases this probability to 25%. This will be the threshold for the experiment at which data loss would be indistinguishable from an observer.

Methodology

- This experiment utilizes four different Bra's as bases, each corresponding to either the **+** or **-** base.
  + : 0° & 90° polarizations **X :** -45° & 45° polarizations
- Either base can represent a binary bit using a specific angle of polarization:
  0 : 90° & -45° polarizations **1 :** 0° & 45° polarizations
- Due to a lack of polarizing lenses, this is simulated computationally on an Arduino UNO board.

The cryptographic algorithm utilized is relatively simple, a single use key is generated consisting of binary digits the same length as the message. If the message were to be transmitted, binary addition of the key to the message would be applied by Alice for encryption. Subsequently, Bob would again apply binary addition of the key to the encrypted message to decrypt it. This would be guaranteed to be safe due to any observer being detected when generating the key. For each trial, distance between transmitter (Alice) and sensor (Bob) is increased. Intervals used: 1m, 2m, 3m

Alice			Bob				Basis Match
Basis	Angle	Bit	Basis	Angle	Detect 0	Detect 1	
+	90°	0	+	0°	100%	0%	Yes
+	0°	1	+	0°	0%	100%	Yes
x	45°	1	+	0°	50%	50%	No
x	-45°	0	+	0°	50%	50%	No
+	90°	0	x	45°	50%	50%	No
+	0°	1	x	45°	50%	50%	No
x	45°	1	x	45°	0%	100%	Yes
x	-45°	0	x	45°	100%	0%	Yes

Figure 2: Table showing detection rate for a given pair of bases

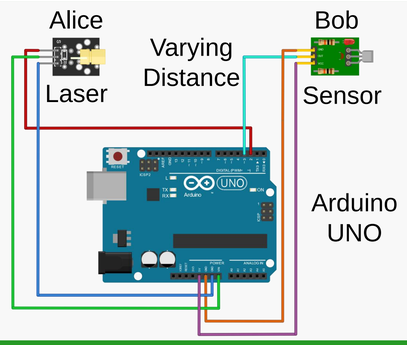

Figure 1: Schematic of experimental setup

Computation

- Alice generates a random set of bases (0°, 90°, -45°, or 45°) and corresponding bits. 52 bases were generated for a 20-bit message to ensure a long enough key.
- Bob generates a random set of bases (0° or 45°).
- Alice then transmits each bit via pulsed laser using the generated base to Bob.
- Bob records the bit received, if the base does not match or there is signal loss a random bit is chosen, this simulates the 50% probability in a quantum system
- Alice and Bob then compare their bases and keep bits with matching bases to be used as the key.
- If only ~25% of bases and bits match, there is an eavesdropper or signal loss so significant as to be indistinguishable, and the key is discarded.

Analysis & Conclusions

Each trial had 100% signal receipt, and an expected ratio of valid base pairs around 50% when generating the encryption key. This indicates that transmission is extremely stable over short distances, even with a low power light source. There are several avenues I would like to explore in the future: increase the distance of transmission, physically polarize the light, use fiber optic cables, and use a trapped photon source instead of laser.

Alice Bases	Bob Bases	Alice Bits	Key Bits Gen.
-45°	45°	0	0
45°	0°	1	1
90°	0°	0	0
0°	0°	1	1
-45°	45°	0	0
0°	45°	1	1
0°	0°	1	1
-45°	45°	0	0
-45°	45°	0	0
0°	0°	1	1
-45°	45°	0	0
45°	45°	1	1
0°	45°	1	1
-45°	0°	0	1
0°	45°	1	1
45°	0°	1	0
0°	45°	1	1
90°	45°	0	0

Results

Sample output using a 2-bit message and 18-bit basis selection. Base pairs that did not match are highlighted in red.

Laser detections	18/18	100%
Valid base pairs	10/18	56%
Valid base/key index	0	2
Valid bases	-45°	90°
Key bits	0	0

Acknowledgements

Thank you to Professor Tom Fleming for being supportive and helping me obtain the necessary materials for this experiment, and to the Edmonds College Foundation for their generous support.

Contact

Jai Veilleux – […]
GitHub - https://github.com/Didgety/