Quantum Cryptography

Julia Wei and Cady van Assendelft
(PHYS 382L)
(Dated: February 21, 2018)

I. INTRODUCTION

To encrypt messages via the one-time pad, in which a random key is shared between the sender and recipient to provide perfect secrecy, secure key distribution is critical. Quantum key distribution (QKD) provides a means to exchange a key between two distant partners that is unconditionally secure [1]. Implemented using a photonic setup, the sent photons constitute the key.

A. Background and History

Introduced in the 1980s, quantum cryptography provides a new method of generating shared secret binary sequences to provide communications security [2]. In this scheme, binary messages are securely encoded using a quantum cryptographic key; the key and message must be the same length.

Quantum key distribution, which is implemented in this experiment, offers security based on the intrinsic properties of photons. QKD offers advantages over current asymmetric encryption, which is currently widely used for both data transmission and authentication. Asymmetric encryption relies on the difficulty of solving certain mathematical problems such as prime-number factorization, problems that quantum computers would be capable of solving exponentially faster than current classical computers. Additionally, as a cryptographic key is only secure for one use, quantum cryptography helps solve the issue of creating large amounts of key material.

QKD has been demonstrated at distances over 1.6 km via an atmospheric optical path [2], over 400 km using optical fiber [3], and over 1200 km in satellite-based entanglement schemes [4].

B. B92 Protocol

One popular implementation for quantum key distribution, and the setup used in this lab, is the B92 protocol. Developed by Charles Bennett in 1992 [5], this protocol uses two photon quantum states that are non-orthogonal. The sender, Alice, encodes a binary bit (1 or 0) as a photon polarized as $|\ 45°\rangle$ and $|\ 0°\rangle$ respectively. The recipient, Bob, randomly measures with the bases $|\ 90°\rangle$, which has a 50% chance of detecting a photon sent in the 0 configuration, or $|\ -45°\rangle$, which has a 50% chance of detecting a photon in the 1 configuration. Because Bob does not know the photon configuration Alice has sent before choosing a measurement basis, there is a probability of 1/2 that the chosen basis is sensitive to the sent photon polarization. This brings the total maximum efficiency to 25%, not including data lost to external sources such as eavesdroppers.

Once Bob has received a string of bits, Bob communicates the time stamps of the bits where he had a detection so Alice knows which bits of her transmitted key she should keep. Once this occurs, Alice and Bob ideally have a shared key that consists of the same random string of bits.

Now suppose we have an eavesdropper, Eve, who manages to obtain partial strings of the bits sent by Alice. For example, Eve can measure photons when Alice sends more than one photon in the same configuration. Secondly, Alice and Bob may not have the same key due to bitflip errors arising from noise in the environment. As a result, Alice and Bob must perform an error reconciliation protocol to reach the same shared key, without giving Eve additional information about their key.

C. Goals

The goal of this experiment is to maximize the final key length while eliminating the risk imposed by eavesdroppers. This includes minimizing the probability of multiple photons per laser pulse, developing an error reconciliation scheme to eliminate discrepancies in Alice's sent and Bob's received key, applying a randomness extractor, and hashing the final key to account for any leaked information.

II. EXPERIMENTAL METHOD

A. Experimental setup

Due to the Heisenberg uncertainty principle, it is impossible to create a laser pulse containing a single photon. Instead our transmitter uses a coherent state. This introduces the security risk associated with the possibility of multiple photons. Using fixed attenuators that decrease the photon number by a factor of 10^9 and crossed polarizers, the average photon pulse was chosen to be on average one or less. In this setup, the transmitted intensity, given in terms of the average number of photons per pulse, $\bar{n}$ is

$$\bar{n} = n_0 \cos^2(\phi - \phi_1) \cos^2(\phi - \phi_2) + n_b$$

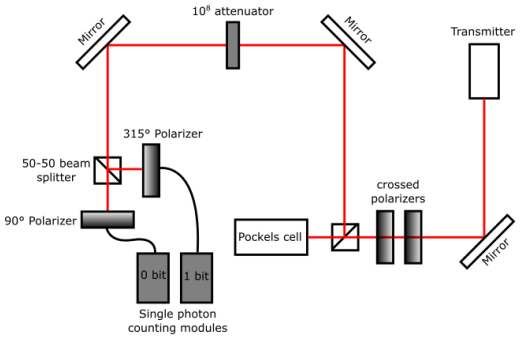

FIG. 1: Experimental setup. The transmitter and two single photon counting modules are connected to a computer, which creates and sends Alice's key, receives Bob's key, and compares the two for bit flip errors.

where n_0 is the number of photons in the attenuated pulse. The angle of the first polarizer relative to the polarization of the laser is given by ϕ, though a correction ϕ_1 is included to account for an offset angle between the laser polarization and the angular scale on the first polarizer. The possible offset angle between the angular scales of the first and second polarizers is given by by ϕ_2. An addition term, n_b is added to the intensity to account for background noise, which consists of stray photons from the room lights as well as dark counts on the single photon detectors.

After passing through the crossed polarizers, the beam then passes twice through a Pockels Cell, which is an electrically activated half-wave plate that rotates a 0° photon to 45°; the Pockels Cell creates the pattern of 1 and 0 bits via voltage inputs.

Once it has passed through the fixed attenuators mentioned above, the beam passes through a $50\%-50\%$ beam splitter which is the implementation of Bob's random basis selection. The beam splitter sends photons randomly to one of the linear polarizers that are described above in section I B. The photon is then detected via a Single Photon Counting Module (SPCM). Pulses are processed only during a small window which is appropriately delayed in coincidence with the laser pulses sent. This means each pulse is sent and received in sequence with a time stamp, allowing Bob to communicate to Alice the time intervals in which he successfully measured a photon.

B. Error Reconciliation Protocol

In order to generate a key, we must perform error reconciliation to remove discrepancies between Alice's sequence and Bob's sequence. However, because the error reconciliation protocol is communicated across a classical channel, it must minimize information leakage.

We first implemented the binary search reconciliation protocol with random shuffling. Then we applied a randomness extractor to remove asymmetry in the total number of 0s and 1s in the sequences. Finally, because Eve may have measured segments of the key, we hash the sequences to produce the final key.

1. *Binary Search and Shuffle function*

The binary search reconciliation protocol with shuffling is described below [6]. The protocol compares the parity of blocks of length P ("P-blocks") in Alice and Bob's sequences. The number of iterations of the protocol is "iter."

0. Initialize iter, P
1. For i in (1:iter)
   a. Find number of P-blocks, `num`
   b. For j in (1:`num`)

      Conduct parity check on Alice and Bob's respective blocks.

      If they match, remove the same bit from each block. Else, conduct a binary search to find an erroneous bit and remove it. For each iteration of the binary search, remove one bit as well.
   c. Generate a random shuffling order, which is used to shuffle Alice and Bob's sequences.
   d. Double P

2. *A simplified parity-based reconciliation protocol*

A simplified reconciliation protocol can use parity checking without the binary search, as shown below.

0. Initialize iter, P
1. For i in (1:iter)
   a. Find number of P-blocks, `num`
   b. For j in (1:`num`)

      Conduct parity check on Alice and Bob's respective blocks.

      If they match, remove one bit from Alice and Bob's sequences. Else, remove all the bits in the P-block from the sequences.
   c. Rudimentary shuffle that moves the end bit of Alice and Bob's sequences to the start.

3. *Randomness Extractor*

For a binary key, even a slight asymmetry in the number of 0s and 1s can reveal information about the encrypted message. Thus, we need a randomness extractor

that will take Alice's and Bob's sequences as input, and produce the respective sequences with an even distribution of 0s and 1s. We used the steps described below:

0. Calculate q, the percentage of 1s in Alice's sequence of length N, $\{a_n\}_{n=1}^N$.
1. Generate a public random sequence of numbers $\{x_n\}_{n=1}^N$ where $x_n \sim unif(0,1)$.
2. If $q < 0.5$, we set $a_n = 1$ when $a_n = 0$ and $x_n < \frac{|q-0.5|}{q}$. Thus, a fraction $\frac{|q-0.5|}{q}$ of the 0s in the sequence are randomly changed to 1s. Else (for $q >= 0.5$), we set $a_n = 0$ when $a_n = 1$ and $x_n < \frac{|q-0.5|}{q}$. Thus, a fraction $\frac{|q-0.5|}{q}$ of the 1s gets changed to 0s. Then there will be equal numbers of 0s and 1s in Alice's sequence.
3. Since $\{x_n\}_{n=1}^N$ is public, Bob can repeat steps 0 and 2 on his sequence $\{b_n\}_{n=1}^N$ to achieve equal numbers of 0s and 1s as well.

If Bob's and Alice's sequences were equivalent to start with, they would remain equivalent after the randomness extractor. In addition, Alice and Bob need not share their own sequences to perform this protocol, and the public random sequence does not reveal anything about their sequences either. Thus, the randomness extractor maintains their privacy.

C. Privacy Amplification

Given that multiple photons can be emitted in a pulse, Eve may receive bits of the sequence initially sent by Alice. To ensure that the quantum key is private, Alice and Bob encrypt their own sequences using a public scheme, yielding their quantum keys. However, when Eve applies the public encryption technique, any discrepancy in Eve's sequence will yield an encrypted key that is completely different from Alice's and Bob's. Thus, Alice and Bob will have private quantum keys.

The encryption is achieved by multiplying a sequence by a publicly available matrix M, an $K \times N$-dimensional matrix where entries M_{ij} are uniformly drawn from $\{0,1\}$. N is the length of the sequence and $K = P(n \leq 1)N$. Then for a given sequence x, the corresponding key will be Mx. M acts as a hash function, in which small changes in an input sequence result in completely different outputs.

III. DATA ANALYSIS AND RESULTS

A. Optimal Polarizer Angle

The main method of attenuating the transmitted laser beam is by crossed polarizers as mentioned in Section II A.

We first measured the intensity of the transmitted laser pulses (in kHz) as a function of the polarization angle given by the first polarizer. The sent frequency of laser pulses is approximately 5 kHz. The intensity function is given by $f(x) = f_0(1 - P(n = 0))$, where f_0 is the frequency of sent pulses, $P(n = 0)$ is the probability of having no photons in a pulse, and x is the polarization angle given in degrees. Given that $\bar{n}$ is the average number of photons in a pulse, the probability of having n photons in a pulse is given by the Poisson distribution

$$P_{\bar{n}}(n) = \frac{\bar{n}^n e^{-n}}{n!} \text{ and } P(n=0) = P_{\bar{n}}(0) = e^{-\bar{n}} \quad (1)$$

We now have the equation $f(x) = f_0(1 - e^{-\bar{n}})$, where

$$\bar{n} = n_0 \cos^2(\pi x/180 - \phi_1)\cos^2(\pi x/180 - \phi_2) + n_b \quad (2)$$

We used Matlab to fit our measured intensity data to $f(x)$ and extract the parameters f_0, n_0, n_b, ϕ_1, and ϕ_2. The fit is plotted in Fig. 2.

Intensity function fit		
parameter	value	95% confidence bound
f_0	4.935	(4.851, 5.018)
n_0	3.941	(3.612, 4.269)
n_b	0.002275	(-0.006676, 0.01123)
ϕ_1	-6.167	(-6.366, -5.937)
ϕ_2	6.05	(5.963, 6.137)

TABLE I: The fitted parameters for the intensity function and their confidence bounds.

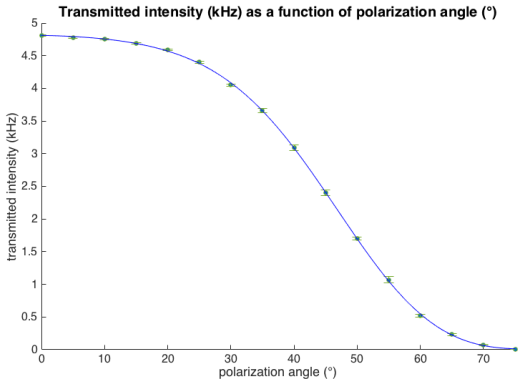

FIG. 2: The intensity of the transmitted pulses is plotted against the polarization angle, and the uncertainty is given by the observed range of the intensity at each angle. The fitted curve is also plotted.

We first calculated

$$\chi^2 = \sum_{n=1}^{N} \frac{(y_i - f(x_i))^2}{\sigma_{y_i}^2}$$

, where N is the total number of different polarization angles at which we measured the intensity. Then angle x_i corresponds to measured intensity y_i and the measured uncertainty σ_{y_i}. (We measured σ_{y_i} by taking the range of values that y_i fluctuated between at each x_i.) Then we get $\chi^2 = 58.22$ and $\chi^2_5 = 11.64$ for the five degrees of freedom in the fit. The reduced value exceeds 1, yet the fit appears to be in good agreement with the data; thus, we may have underestimated the σ_{y_i}. Next, we calculate χ^2 based on the expected variance in the intensity y_i at each angle x_i. It is given by $var(y_i) = P(n = 0)(1 - P(n = 0))f_0$, where $P(n = 0)$ is a function of x_i. Using

$$\chi^2 = \sum_{n=1}^{N} \frac{(y_i - f(x_i))^2}{var(y_i)}$$

yields $\chi^2 = 0.0085$ and $\chi^2_5 = 0.0017$, so the data agrees with the fit when we use the expected uncertainties for y_i.

We can also plot $\sqrt{var}$ (standard deviation) against the measured ranges of the intensity given by σ_{y_i}.

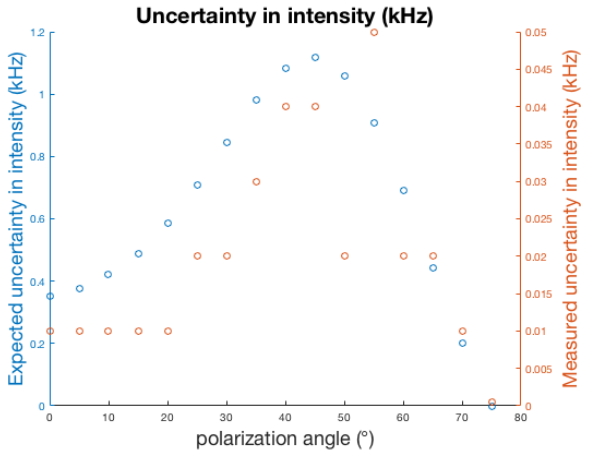

FIG. 3: The expected standard deviation in intensity is plotted in blue, while the measured fluctuation ranges of the intensity is plotted in orange.

The shape of the expected and measured curves appear to be in agreement in Fig. 3, so our measured uncertainties resemble the $\sqrt{var}$ of a Poisson process. This supports the fact that the number of photons in a pulse follows the Poisson distribution. However, the values do not agree. This could be because the standard deviation isn't a good explanation for the range of values. It's also possible that we didn't observe the intensity values long enough at each polarization angle to measure a truly representative range of values.

In order to determine the optimal polarization angle, we can set $1 - P(n = 0) = 0.25$, because we are interested in achieving 25% efficiency for the transmitted signal. Then, using our parameters in Table I, we can solve for the optimal angle x_{opt}. We find $x_{opt} = 53.48°$. The 95% confidence bound is (45.28, 61.66), where the lower and upper bounds for x_{opt} are found by using the lower and upper parameter bounds in Table I, respectively. Additionally, we note that using x_{opt} does not imply that $\bar{n} = 1$, as is expected in the ideal case for QKD efficiency; we actually have that $\bar{n} = 0.30$.

B. Optimal Parameters for Error Reconciliation Protocol

1. *Block Size*

We based our P-block size on the reference model of the Cascade protocol cited by Martinez-Mateo et al. [6]. In that Cascade protocol, the initial block size is determined to be $0.73/Q$ where Q is the error rate. Although our error reconciliation protocol varies somewhat from cascade, as it implements the same type of binary search we used this calculation as a starting point for determining our block sizes.

The error rate varied due to environmental conditions, so we used both a high and low estimate of the error rates (2% and 1.1% respectively) to find two block sizes using the formula above and rounding to the nearest power of two. We tested our error reconciliation protocol both with initial block sizes of 32 and 64 bits, and found empirically that an initial block size of 32 maximized error removal while resulting in minimal data loss. After each iteration of the error reconciliation protocol, the block size doubles.

2. *Iterations*

With each iteration of the error reconciliation protocol, data is lost as Alice and Bob need to throw out a bit for each parity check, as well as throwing out the bits with errors. However, each iteration removes more errors; errors are missed in the first passes if the number of errors in a P-block is even. The optimal number of iterations minimizes the bit loss while ensuring all errors are removed.

Empirically, we found the minimum number of iterations needed for the binary search and shuffle protocol was four. This matches with the result cited for Cascade protocol in Martinez-Mateo et al. [6]. Although this was higher than the iterations necessary for a pair-wise or 3-bit parity check protocol, the loss rate was considerably lower. The total necessary iterations and bit loss rates for each error reconciliation protocol are summarized in Table II, and the error and bit loss rates after each iteration are shown in Figure 4.

3. *Hash Function*

To find $K = P(n \leq 1)N$, we calculated $P(n \leq 1) = (1+\bar{n})e^{-\bar{n}}$ using the parameters in Table I and x_{opt}. This

Protocol	Iterations required	Total percent loss
Pair-wise parity check	2	75.54%-75.60%
3-bit parity check	3	71.23%-71.37%
Binary search	4	9.20%-9.76%

TABLE II: Iterations required (to assure error rate reached zero) and average loss rate (after all iterations) for each of three error reconciliation protocols tested.

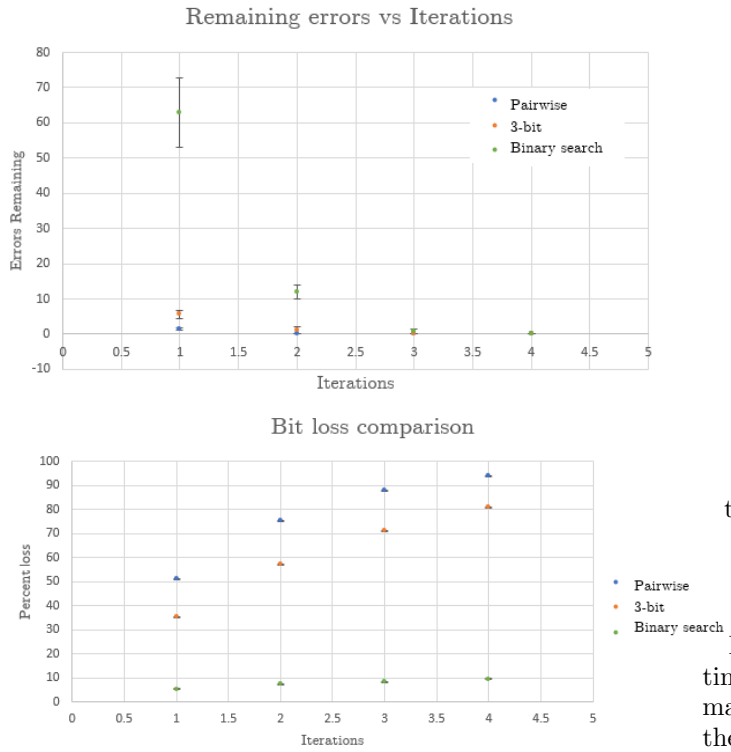

FIG. 4: Remaining errors and bit loss after each iteration, shown for pair-wise, 3-bit, and binary search and shuffle function protocols. The uncertainty is given as the standard error in the percent loss and errors remaining.

yielded $P(n \leq 1) = 0.967$ with a 95% confidence bound of (0.966, 0.969). Applying the hash function to the key reduces its length by 3.3% to achieve the final key length.

C. Optimal environmental conditions

Environmental conditions such as the room light settings had an impact on the efficiency of data transmission and error rates. To examine the impact of room lights, we assigned each setting on the light panel an integer from 0 (completely off) to 7 (brightest setting). The bitflip error rate was the lowest when the room lights were turned completely off; in Fig. 5, the error rate given by the number of bitflips seems to scale linearly with the light level. With lights higher than mid-brightness (level 4), the number of errors was too great for the standard 4-iteration error reconciliation protocol to fully eliminate.

We chose the lowest light setting (1) to be the default operating condition; all trials were conducted with this light setting unless otherwise explicitly noted.

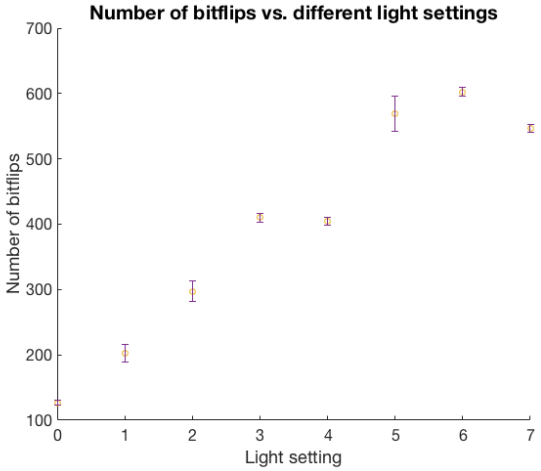

FIG. 5: We report the number of bitflips in the transmitted bit sequence, along with standard error. Three trials were performed for each light setting.

In addition to calculating the impact of the light settings on the transmission and error rate, we also estimated the background noise rate, n_b directly by turning the laser off and recording the transmitted intensity for each channel. The results, measured in kHz, are given in Table III.

Background noise by light level		
Light setting	Channel	Intensity
Off	1	0.002 ± 0.0005 kHz
	2	0.001 ± 0.0005 kHz
Low	1	0.006 ± 0.002 kHz
	2	0.006 ± 0.002 kHz
High	1	0.03 ± 0.01 kHz
	2	0.02 ± 0.005 kHz

TABLE III: Estimated background noise n_b for each channel under varied light settings. The laser was turned off but the laser pulse gate was kept on, meaning the signal was only measured during a window of time equivalent to the duration of the laser pulse sent.

Additional environmental conditions such as humidity and temperature may have had an impact on the number of transmitted bits or the number of errors; however, these impacts were neither analyzed nor controlled for during the experiment.

D. Overall efficiency

To test the overall efficiency of the setup as well as other parameters including the asymmetry, we ran 30 trials with optimal error reconciliation parameters and light settings.

The overall efficiency was measured by dividing the final key length (after the binary search and shuffle function and hash function) by 25,000, which is the theoretical maximum key length for perfect, error-free transmission.

The data loss rate takes into account only the amount of bits lost during the binary search and shuffle function, and not the additional 3.3% reduction in key length due to privacy amplification.

The results, summarized in Table IV, showed that the overall efficiency averaged just over 85%. The error rate averaged to 1.1%. Additionally, the average asymmetry (ratio of 1 bits to 0 bits) was 0.554. This could be attributed to Bob simply having a more efficient detector for Channel 1, the channel that measures 1 bits.

Value	95% Confidence bound
Overall efficiency	(84.81%, 85.80%)
Error rate	(0.95%, 1.28%)
Data loss rate	(9.36%, 9.55%)
Asymmetry	(0.551, 0.558)

TABLE IV: Confidence bounds for mean values based on a sample of 30 trials with optimal operating conditions.

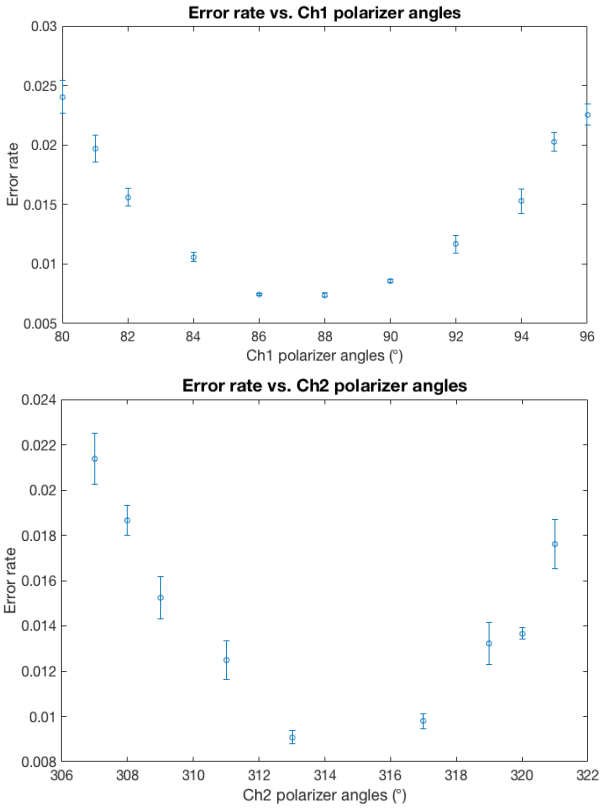

FIG. 6: The error rate (ratio of bitflips to the total number of received bits) as a function of Channel 1 and 2's polarizer angles.

E. Robustness against misaligned polarizers

To test the robustness of the error reconciliation protocol, we purposefully misaligned one or both of Bob's polarizers. This corresponded to an error in his measuring bases.

Each individual polarizer could handle about $\pm 6°$ degrees from the intended angle. For Channel 1, the error reconciliation protocol could remove all errors (within eight iterations, doubled from the normal four)for a range of polarization angles from $83° - 96°$. For Channel 2, the range of polarization angles the error reconciliation protocol could correct completely for was $308° - 320°$. The results of this are shown in Table V.

We also plotted the error rate as the ratio of the number of bitflips over the total number of received bits in Fig. 6. We note that the error rate does not reach a minimum when the Channel 1 polarizer is at 90° (as the minimum appears to be around 88°). This may be because the 90° marking on the polarizer does not completely correspond to the polarizer's rotation at 90° to the horizontal.

As seen in Table V, with 8 iterations our error reconciliation protocol could successfully remove all errors even when the error rate was twice the average value in optimal conditions.

Polarizer offset	Error rate	ER Protocol Results
$-10°$	0.030	Failed
$-7°$	0.020	Succeeded
$-5°$	0.017	Succeeded
$0°$	0.012	Succeeded
$5°$	0.022	Succeeded
$7°$	0.0276	Failed
$10°$	0.037	Failed

TABLE V: Error rate for misaligned polarizer configurations. Error rates averaged across trials with the same offset, regardless of which channel's polarizer was misaligned. "Failed" indicates errors remained after 8 iterations; "Succeeded" means all error were removed.

Additionally, we examined the effects of misaligning both polarizer angles simultaneously; the error reconciliation protocol failed to remove all errors only when the polarizers were each misaligned by more than $+2°$ or $-6°$.

Another interesting trend was that the total number of bits received increased as the polarizer angle increased rather than being symmetric about the correct polarizer angle like the error rate was.

IV. CONCLUSION

In addition to achieving an average of 85% overall efficiency with optimal operating conditions and parameters, our error reconciliation protocol was able to handle artificially increased error rates due to misaligned receiving polarizers and increased room lighting.

One way to improve this setup further would be to increase the efficiency of the error reconciliation protocol. Implementing Cascade would lower the data loss rate, as that protocol additionally keeps a history of previous errors removed in order to account for issues such as double errors in a P-block. Additionally, the efficiency of the hash function could be improved; currently, the hash function iterates over each column in the matrix containing the final key instead of multiplying the key by a single $K \times N$ dimensional matrix. This was necessary because the current version of MatLab on the lab's computer cannot store a matrix that large.

Additional improvements to this setup include further investigation of other environmental conditions including room temperature and humidity, and how they impact the transmission rate and bitflip error rate.

[1] S. Nauerth *et al.*, Nature Photonics **7**, 382 EP (2013).
[2] W. T. Buttler, R. J. Hughes, S. K. Lamoreaux, G. L. Morgan, J. E. Nordholt, and C. G. Peterson, Phys. Rev. Lett. **84**, 5652 (2000).
[3] H.-L. Yin *et al.*, Phys. Rev. Lett. **117**, 190501 (2016).
[4] J. Yin, Y. Cao, *et al.*, Science **356**, 1140 (2017), http://science.sciencemag.org/content/356/6343/1140.full.pdf.
[5] C. H. Bennett, Phys. Rev. Lett. **68**, 3121 (1992).
[6] J. Martinez-Mateo, C. Pacher, M. Peev, A. Ciurana, and V. Martin, Quantum Info. Comput. **15**, 453 (2015).