An Information-Theoretically Secure QKD Protocol for One-Time Pad Encryption

Sergejs Kozlovičs[▽ [0000−0002−7085−383X]], Krišjānis Petručeņa[0009−0008−5713−5914], and Juris Vīksna[0000−0003−2283−2978]

Institute of Mathematics and Computer Science, University of Latvia
Raiņa bulv. 29, Riga, Latvia, LV-1459
▽ Corresponding author: sergejs.kozlovics@lumii.lv

Abstract. Quantum Key Distribution (QKD) protocols are designed to establish a shared secret key by combining classical and quantum mechanical methods, guaranteeing security even against adversaries with unlimited computational power. However, all widely-known QKD protocols require an authenticated classical channel, which is usually implemented using a pre-shared key (PSK) and a stream cipher, meaning that QKD protocols are *not* information-theoretically secure (ITS). In this paper, we propose a novel *ITS* QKD protocol, which generates a shared *authenticated* bitstream, which can be safely used for OTP encryption.

Keywords: quantum key distribution · QKD · one-time pad· OTP · information-theoretical security · ITS

1 Introduction

Quantum Key Distribution (QKD) is a secure key agreement method involving classical communication and quantum mechanics.

Quantum Mechanics and strong probabilistic arguments allow us to detect if somebody has eavesdropped on or tampered with the key.[1]

The first QKD protocol, BB84, has been known since 1984, with the security proof appearing only in 2000 [3, 6]. Other popular protocols are B92, SARG04, the Ekert protocol [2, 5, 4]. Some are prepare-and-measure protocols, some are entanglement-based, and some have both variations.

All such protocols require an *authenticated* classical channel; otherwise, a man-in-the-middle attack is possible. For authentication, a pre-shared key (PSK) is usually configured at both QKD endpoints (Alice and Bob). Thus, QKD can be considered a process of extending the initial PSK to a potentially infinite stream of shared bits, which have not been altered or eavesdropped on.

However, a PSK is needed not only for the initial authentication but also for encrypting and/or validating protocol-specific classical communication. Here we face an intrinsic "chicken and egg problem": the number of bits transmitted via

[1] Eve=Mallory in the quantum setting since measurements impact quantum states.

the classical communication exceeds the number of bits that will be generated by a QKD protocol.[2]

In modern QKD deployments, this problem is swept under the rug using a stream cipher (such as AES in GCM mode) for the classical channel. If the QKD key is also aimed at being used in a stream cipher, that imposes no additional risk, provided that sessions of the classical part of QKD are short. (The latter can be done by re-initializing and re-authenticating the classical QKD link by sacrificing one QKD key for a new PSK.)

However, for information-theoretical secure (ITS) applications requiring OTP[3] encryption, the QKD protocol itself must also be ITS. Is it possible to design a QKD protocol without relying on a stream cipher, while still providing an authenticated classical channel secured with OTP? We answer positively by introducing the KPV25 protocol.

2 The KPV25 Protocol Prerequisites

The KPV25 protocol is an ITS QKD protocol with built-in authentication. It is a prepare and measure, discrete variable protocol, which uses the well-known BB84/Wiesner qubit states: $|0\rangle, |1\rangle, |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. We denote the rectilinear (or computational, or orthogonal) basis as $R = \{|0\rangle, |1\rangle\}$. We denote the diagonal (or Hadamard) basis as $D = \{|+\rangle, |-\rangle\}$.

2.1 Prerequisites

First, the protocol relies on *Quantum Mechanics* stating that it is impossible to distinguish between two orthogonal states if they are measured in the incorrect basis (D vs. R), which forces the state to collapse into one of the measurement basis states with probability $1/2$.

Next, we assume that Alice and Bob are equipped with two *untrusted channels*: a bi-directional classical channel and a unidirectional discrete quantum channel for transmitting single qubits (polarized photons over a dark fiber are good for that).[4]

We also assume that the quantum channel is *synchronized*, i.e., for each discrete time period ΔT Alice encodes and sends n qubits, and, if Eve is not present, Bob detects a qubit correctly for each time interval $\frac{\Delta T}{n}$ with probability $1 - \varepsilon$.[5]

[2] For example, the sifting phase of BB84 transmits const$\cdot n$ classical bits to generate less than $n/4$ key bits.

[3] one-time pad, also known as Miller telegraphy system and Vernam cipher

[4] A KPV25 variant for the bi-directional quantum channel is also possible. It achieves a higher bitrate but requires two quantum channels, doubling the infrastructure costs.

[5] For a photon-based channel, that means allowing false photon detections or missing photon substitutions with some default value (say, $|0\rangle$) $\varepsilon \cdot 100\%$ of time.

Finally, we assume that Alice and Bob had agreed on some *error-correcting code (ECC) and a checksum (hash) function* with the parameters chosen in accordance with the KPV25 parameters (512 and 16 in the example below). We do not rely on the strength of hash functions, but they should allow for the detection of invalid sequences with high probability.

2.2 The Protocol

Unlike in other discrete-variable prepare-and-measure QKD protocols, Alice and Bob will know measurement bases in advance: they will never have to guess. Besides, bases are never revealed publicly. Eve *can* guess the bases and impact Bob's measurements; however, as we will see, if Eve impacts too many bits, the key will be discarded.

The protocol consists of a potentially infinite number of *iterations*.

Protocol iteration input: a 512-bit PSK.[6]

Protocol iteration output: a newly generated PSK_{new} of length $512+2q$ bits. Thus, we receive a PSK_{new} for the next iteration as well as $2q$ "user bits", which can be used as OTP for user data. We use $q = 16$ below.

Protocol iteration:

1. Alice uses the PSK to choose the bases: base R for PSK bit 0 and base D otherwise.
2. Alice generates $256 + q = 256 + 16$ random bits and computes the corresponding 128-bit hash (a 128-bit MD5 or a truncated SHA-3 output can be used for the hash.)
3. Alice adds 112 more bits to obtain a 512-bit codeword according to the chosen ECC. For 400 data bits (256+16 bits and 128 hash bits), we can use the Reed-Solomon code RS(64, 50) with 8-bit RS symbols. Such a code can correct up to 7 RS symbols (up to 7 bits in the worst case and up to 56 error bits in the best case), which corresponds to ε between 0.0175 and 0.14 (the error rate of the quantum channel).
4. Alice uses the quantum channel to send the 512-bit codeword in qubits encoded in bases chosen at Step 1: 0 is encoded as $|0\rangle$ or $|+\rangle$, while 1 is encoded as $|1\rangle$ or $|-\rangle$ (as in classical QKD protocols).
5. Bob receives 512 qubits and measures them all in the correct bases since he knows the PSK. Let's denote measurement results $a_1, a_2, \ldots, a_{512}$. Then, Bob uses ECC to correct possible errors (in-place).
6. Bob checks whether $hash(a_1 \ldots a_{272}) = a_{273} \ldots a_{400}$. This check ensures that
   - Alice knows PSK.
   - Eve has no or negligible information about the PSK.[7]

   Bob discards the iteration if the check does not succeed (Bob informs Alice about that in Step 7).

   Since the PSK was used only quantumly to encode the bases (which Eve had to guess), Bob can re-use the same PSK classically when replying to Alice!

[6] We fix the constant 512 to simplify the narrative. All constants can be parametrized.

[7] Due to imperfect photon sources, Eve can have *some* information about the PSK, see Section 3.

7. Bob also generates 256+16 random bits and computes the corresponding 128-bit hash. If Bob has discarded the protocol iteration in Step 6, he forces all 400 bits to be all zeroes (a special value). We assume that $hash(0^{272}) \neq 0^{128}$; thus, Alice can distinguish between an error and all zeroes generated by accident.
8. Bob adds 112 more bits to obtain a 512-bit ECC codeword.
9. Bob uses the same PSK as Alice to send the OTP-encrypted codeword to Alice via the classical channel. Bob forgets the PSK, ensuring the OTP security of each protocol iteration.
10. Alice receives 512 bits $x_1, x_2, \ldots, x_{512}$. She uses ECC to correct possible errors (in-place) and then decrypts Bob's bits $b_1, b_2, \ldots, b_{512}$ (using the PSK, which Alice then forgets).
11. Alice checks whether $hash(b_1 \ldots b_{272}) = b_{273} \ldots b_{400}$. Thus, Alice can be sure that the message originated from Bob (since only he knew the PSK) and that Eve has negligible information about the received bits. If the check does not succeed, Alice discards the protocol iteration. Alice *does not* inform Bob about that (see below).

Output: $\mathrm{PSK}_{new} = a_1 a_2 \ldots a_{256} b_1 b_2 \cdots_{256}$;
$2q$ user bits are $a_{257} \ldots a_{272}\ b_{257} \ldots b_{272}$. Since 32 user bits are returned here, we need 16 iterations to obtain a full 512-bit user key.

Since a protocol iteration can be discarded in Line 6 or 11, Alice and Bob maintain a buffer of recovery PSKs, which is filled gradually by sacrificing some user bits. Each recovery key is used at most once as a PSK replacement for the next protocol iteration.

Notice that if Bob discards the protocol, so does Alice. If Alice discards the protocol (by herself or because of Bob), she takes the oldest non-used recovery PSK from the buffer for the next protocol iteration. How can Bob find out that Alice has discarded the last protocol iteration if Bob hasn't discarded it? At the next step, Bob's check will fail (since Alice used the recovery PSK instead of PSK_{new}). In this case, Bob also tries the oldest non-used recovery PSK and finds out whether he should switch to the recovery PSK.

3 Preliminary Analysis and Safety Against PNS Attacks

Since the measurement bases are never revealed publicly, any measurement-based attack requires guessing, which leads to incorrect measurements by Bob in 25% of the time. For a 512-bit case, these are 128 bits, up to 56 of which Bob can recover but the other 72 bits give high probability of the iteration to be rejected due to an invalid checksum.

However, since true single-photon sources are hard to implement, Eve can use a Photon Number Splitting (PNS) attack to capture some of the equally-encoded photons from multi-photon pulses [1]. How much information can she get without the key being discarded?

We can assume that laser pulses from Alice contain exactly 1 photon at least 50% of the time and contain 3 or more photons in less than 12.5% of the time.[8] Now, Eve can intercept-and-resend only 112 1-photon pulses, which leads to ≈ 56 bits being changed — the maximal amount Bob can recover while still being able to verify the checksum. Eve gets 3/4 of these 112 bits correctly by choosing the measurement bases randomly. From each of the $\geq 37.5\% = 192$ 2-photon pulses, Eve can steal 1 photon, obtaining values for 3/4 of the bits. From the remaining $\leq 12.5\% = 64$ pulses, Eve can steal 2+ photons and get the correct bit with a probability close to 1. By simple Math, that gives Eve ≈ 258 bits of information for each 512-bit user key. We then can use any privacy amplification method. Since bits are independent, by simply XOR-ing 10 user keys, Eve's expectations are only ≈ 0.54 bits for a 512-bit key.

4 Conclusion

The KPV25 protocol is ITS in the sense that each PSK is used only once in classical channels available to the protocol. The key bitstream generated by KPV25 can be applied for OTP encryption. The protocol does not need the traditional bit sifting phase, and error correction is built-in. Parameter estimation is unnecessary since hash checks give a high chance of discarding a key if too many qubits are "stolen"/altered. However, the protocol needs some privacy amplification method (a simple XOR-based method apparently can be used).

A deeper analysis, a security proof, and the study of possible coherent attacks[9] on KPV25 are still needed.

Acknowledgements Research supported by the "Latvian Quantum Technologies Initiative" project No. 2.3.1.1.i.0/1/22/I/CFLA/001 co-funded by the European Union, https://www.quantumlatvia.lu.lv.

References

1. Ashkenazy, A., Idan, Y., Korn, D., Fixler, D., Dayan, B., Cohen, E.: Photon Number Splitting Attack – Proposal and Analysis of an Experimental Scheme. Advanced Quantum Technologies **7**(7), 2300437 (Jul 2024). https://doi.org/10.1002/qute.202300437
2. Bennett, C.H.: Quantum cryptography using any two nonorthogonal states. Physical Review Letters **68**(21), 3121–3124 (May 1992). https://doi.org/10.1103/PhysRevLett.68.3121
3. Bennett, C.H., Brassard, G.: Quantum cryptography: Public key distribution and coin tossing. In: Proceedings of IEEE International Conference on Computers, Systems and Signal Processing. pp. 175–179. Bangalore, India (Dec 1984)

[8] It is the case with truncated Poisson photon number distribution with $\lambda = 1$ and maximal photon number ≈ 10 for 512 photons sent by Alice.

[9] the attacks, where Eve has unlimited computation power and can interact with all quantum states simultaneously

4. Ekert, A.K.: Quantum cryptography based on Bell's theorem. Physical Review Letters **67**(6), 661–663 (Aug 1991). https://doi.org/10.1103/PhysRevLett.67.661, https://link.aps.org/doi/10.1103/PhysRevLett.67.661
5. Scarani, V., Acín, A., Ribordy, G., Gisin, N.: Quantum Cryptography Protocols Robust against Photon Number Splitting Attacks for Weak Laser Pulse Implementations. Physical Review Letters **92**(5), 057901 (Feb 2004). https://doi.org/10.1103/PhysRevLett.92.057901, https://link.aps.org/doi/10.1103/PhysRevLett.92.057901
6. Shor, P.W., Preskill, J.: Simple Proof of Security of the BB84 Quantum Key Distribution Protocol. Physical Review Letters **85**(2), 441–444 (Jul 2000). https://doi.org/10.1103/PhysRevLett.85.441