Chapter 8
The Impact of Key Lengths on QKD Security: An ML Study

Hasan Abbas Al-Mohammed
Qatar University, Qatar

Afnan S. Al-Ali
Qatar University, Qatar

Elias Yaacoub
Qatar University, Qatar

Khalid Abualsaud
Qatar University, Qatar

ABSTRACT

Charles Bennett and Gilles Brassard, in 1984, proposed the first QKD protocol and called BB84. It is assumed the protocol shares a quantum key safely (between two parties). In 2000 it was implemented easily and showed a significant method for detecting an attacker that trying to get the shared key by utilizing the final key length. When the length exceeds a certain value, that is calculated before the transmitting the key, the majority of prior works agree with that, but this chapter showed a significant threat that affects the final key to same. There is no attacker at the middle. Moreover, this chapter analyzed the final key lengths and showed the harmed value of the final key length with the attacker effect. It also showed how often these values could be within the threshold. Furthermore, a solution was found to detect the attacker by using a machine learning technique. The results showed a promising accuracy to detect the attacker relying on the final key length.

DOI: 10.4018/978-1-7998-9522-0.ch008

1. INTRODUCTION

Quantum Key Distribution (QKD) is a technique involving two members (Alice and Bob) to exchange a private symmetric key. If an attacker (Eve) decided to steal the private key in a QKD protocol, communicators can see it using appropriate quantum rules (e.g., the well-known Heisenberg uncertainty theory) (Busch et al., 2007).

The first QKD protocol BB84 was proposed. BB84 is currently the most general and powerful quantum cryptography protocol for transmitting data utilizing photon polarization states. Moreover, as Eve tries to interfere with quantum networks, the protocol detects the attack and stops key generation. Furthermore, the protocol will not be suspended as long as Eve is passive. For any quantum network assault, the likelihood that the protocol does not stop and an attacker duplicates the generated keys is extremely low (Shor & Preskill, 2000).

Alice and Bob run a single search before the shared key stream may be utilized confidently. If they find a large number of mismatches, they publicly select and check few bits at random from their key streams, and if the error rate exceeds a predetermined threshold, they trash the entire shared key and generate a new one. The thresholds rely on the initial photons sent from the transmitter (Alice) in order to obtain the final key lengths used for generating the secret key (Jeong et al., 2020). Moreover, the ratio between the maximum and minimum range of the final key length plays a role in terms of utilizing the key in a given function or encryption algorithm. The effect of raising the initial photons sent by the server on the final key length at the destination needs to be quantified, for example in the case of IoT devices. Therefore, it is important to use several values for the number of initial photons and compare them with the corresponding final key length, in order to demonstrate the effect of increasing the number of initial photons on the final key length for utilizing it for securing the communication or making it hard to detect by an attacker (S. K. Singh et al., 2020).

However, sometimes the attacked final key length remains within the threshold. In this case, the receiver does not know that there is an attacker in the middle trying to detect the key that is shared between the sender and receiver, because the final key length does not exceed the thresholds range that has been agreed between them. In this case, the vulnerabilities of QKD lead to losing the possibility of detecting the attacker and changing the keys. In addition, the percentage for the attacked key within threshold depends on the initial photons. Therefore, the best way to discover an attacker even when the final length is within threshold is to use machine learning techniques (Zbinden et al., 1998).

To achieve this aim, a support vector machine (SVM) classifier is used. For our given problem and the type of our data which consists of one feature represented by the key to each sample, SVM is the most suitable algorithm because it maps the