Linköping studies in science and technology. Theses.
No. 1447

Weaknesses of Authentication in Quantum Cryptography and Strongly Universal Hash Functions

Aysajan Abidin
ئەيساجان ئابىدىن

Linköping University
INSTITUTE OF TECHNOLOGY

Department of Mathematics
Linköping University, SE–581 83 Linköping, Sweden
Linköping 2010

Linköping studies in science and technology. Theses.
No. 1447

Weaknesses of Authentication in Quantum Cryptography and Strongly Universal Hash Functions –

Aysajan Abidin
ئەيساجان ئابىدىن

Abuding.Aishajiang@liu.se
www.mai.liu.se
Division of Applied Mathematics
Department of Mathematics
Linköping University
SE–581 83 Linköping
Sweden

ISBN 978-91-7393-354-4 ISSN 0280-7971

Printed by LiU-Tryck, Linköping, Sweden 2010

To my Mother, Guzelnur, Éhsan and my family.

مەن ئۇشبۇ ئىلمىي ماقالەمنى ئانامغا، گۈزەلنۇرغا، ئېھسانغا ۋە ئائىلەمگە بېغىشلايمەن.

Abstract

Authentication is an indispensable part of Quantum Cryptography, which is an unconditionally secure key distribution technique based on the laws of nature. Without proper authentication, Quantum Cryptography is vulnerable to “man-in-the-middle” attacks. Therefore, to guarantee unconditional security of any Quantum Cryptographic protocols, the authentication used must also be unconditionally secure. The standard in Quantum Cryptography is to use the Wegman-Carter authentication, which is unconditionally secure and is based on the idea of universal hashing.

In this thesis, we first investigate properties of a Strongly Universal hash function family to facilitate understanding the properties of (classical) authentication used in Quantum Cryptography. Then, we study vulnerabilities of a recently proposed authentication protocol intended to rule out a "man-in-the-middle" attack on Quantum Cryptography. Here, we point out that the proposed authentication primitive is not secure when used in a generic Quantum Cryptographic protocol. Lastly, we estimate the lifetime of authentication using encrypted tags when the encryption key is partially known. Under simplifying assumptions, we derive that the lifetime is linearly dependent on the length of the authentication key. Experimental results that support the theoretical results are also presented.

Populärvetenskaplig sammanfattning

Risken för illegal avlyssning av information, till exempel vid penningtransaktioner, tvingar fram allt mer avancerade tekniker för kryptering. När man skickar krypterade meddelanden via datornätverk är ett svårlöst problem hur nyckeln ska överföras. Ett sätt är att skicka den med kurir (vanlig post eller, som i agentfilmer, en person med attachéväska fastlåst vid handleden). En kurir måste förstås vara pålitlig, annars finns risken att nyckeln omärkligt kopieras på vägen. En annan teknik är så kallad öppen-nyckel-överföring som används för Internetbank och säkerhetsfunktioner i webbläsare (https). Öppen-nyckel-överföring anses säker, eftersom det krävs stora beräkningar för att knäcka de långa strängar av databitar (omkring 2 000) som nyckeln består av.

Det finns en ny teknik för att överföra nyckeln som kallas kvantkryptografi där säkerheten garanteras av kvantmekaniska naturlagar. Än så länge är det dock mycket få som använder den. Det behövs en speciell hårdvara med till exempel en typ av laser som sänder ut enstaka polariserade ljuspartiklar (fotoner) via optisk fiber eller genom luften. Några företag och banker i Österrike provar systemet och försök pågår med satellit-tv-överföring. Säkerheten garanteras eftersom kvantmekaniska objekt har den mystiska egenheten att de inte tål att mätas eller manipuleras utan att förändras. Om någon försöker kopiera en kvantmekaniskt kodad nyckel på vägen, så kommer det att märkas i form av brus. En avlyssnare kan ställa till problem, men inte få ut någon användbar information utan att det märks.

Denna avhandling handlar om den del av ett kvantkryptosystem som ska se till att man överför nyckeln till rätt person. Nyligen hittade man en svaghet i det autentiseringssystem som föreslagits för kvantkrypto, det finns en teoretisk möjlighet att en obehörig person kan få ut nyckeln utan att upptäckas genom att samtidigt manipulera både den kvantmekaniska och den vanliga kommunikation som behövs. Avhandlingen behandlar denna svaghet, och dessutom två förenklade system tänkta att öka nyckelproduktionen. Resultaten inkluderar svar på frågor om de olika varianter som finns av kvantkryptografi är olika känsliga, och även råd om säker användning av systemen.

Acknowledgments

I would like to thank my supervisor docent Jan-Åke Larsson for introducing me to this project with great patience. I am especially grateful for all the support, motivation, and encouragement that he has given me. I know that I can never thank him enough, but I can begin thanking him now.

I am grateful to my co-supervisor Associate Professor Viiveke Fåk for proofreading the Thesis and the papers, and for giving me constructive feedbacks.

I would also like to express my appreciation of numerous help and support from the Director of Graduate Studies Dr. Bengt-Ove Turesson, Professor Brian Edgar, and Professor Lars-Erik Andersson.

I must thank all the PhD students in the Mathematics Department here at Linköping University for creating a nice and friendly working atmosphere.

Last but certainly not least, I am deeply indebted to my mother, sisters, brother, and my family–Güzelnur and Éhsan–for always supporting me, believing in me and standing behind me.

مەن بۇ پۈرسەتتىن پايدىلىنىپ مېھرىبان ئانامنىڭ مېنى تەربىيەپ قاتارغا قوشۇش جەريانىدا ماڭا سىڭدۈرگەن چەكسىز ئەجرىگە ۋە ماڭا بەرگەن ئالەمچە مېھرىمۇھەببىتىگە چىن كۆڭلۈمدىن چوڭقۇر تەشەككۈر ئېيتىمەن.

Linköping, June 3, 2010
Aysajan Abidin
ئەيساجان ئابدىن

Contents

1

Introduction and outline

When two parties, which have not had previous contact and are separated far from each other in space, want to communicate with each other secretly, it is impossible for them to achieve this without sharing a string of secret bits. They can either use a courier to send the secret key, or meet in person to exchange keys so that they can send secret messages to each other later on. Both of these are time consuming and expensive.

Public Key Cryptography (PKC) is one solution to this problem. PKC schemes are based on computationally hard[1] problems in number theory such as prime factoring (as in RSA), solving discrete logarithm problems (equivalently known as Diffie-Hellman problem) and so on. The security of these systems is solely built on the (unproven) assumptions that the above mentioned problems are computationally hard to solve using classical computers.

Quantum computing, however, presents quantum Fourier algorithms such as Shor's algorithm [1], which can be applied to solve the factoring problems and discrete logarithm problems efficiently (with polynomial effort) on a quantum computer. This implies that quantum computers, if ever built, can be used to break RSA or Diffie-Hellman cryptosystems. Therefore, unconditionally secure key distribution protocols are needed.

A possible alternative for key distribution is QKD, which is unconditionally secure, and its security is based on the laws of nature, *not* on computational complexity as is the case for classical systems. Since the introduction of the first QKD protocol by Bennet and Brassard [2] in 1984 (BB84) it has widely been studied and big theoretical and technological advances have been made, which led to commercial QKD products manufactured by, for example, idQuantique, based in Geneva. However, the quantum part of QKD is not enough on its own to securely transmit secret keys. Practical implementations require the communicating parties to have an immutable public channel, without which QKD is vulnerable to a man-in-the-middle (MITM) attack. To prohibit such an attack on QKD,

[1] Here computationally hard means that the best algorithm for a problem depends exponentially, in time, on the input size.

proper message authentication is needed. Therefore, QKD is secure only if it is combined with an unconditionally secure message authentication scheme.

The focus of this thesis is on authentication used in QKD. The standard in QKD is to use the Wegman-Carter authentication, which is provably unconditionally secure. It is unconditionally secure in the sense that without the knowledge of the secret key all tag values are equally possible for any given message, and even when a message-tag pair is known all tags are almost equally likely for another message. Therefore, Eve is not in an improved position even after seeing a valid message-tag pair. An arbitrarily small security threshold, in the form of a low probability of Eve being able to calculate the valid tag for any forged message after seeing a valid message-tag pair, can be obtained by choosing an appropriately long tag length.

There are, however, two things that need be taken care of. One, what happens when the authentication key is partially known? In [3] and [4], the authors studied security of the Wegman-Carter authentication in the context of QKD. They showed that the Wegman-Carter authentication becomes sensitive to the choice of messages if the key is not completely secret. Also, they proposed a simple solution to this problem. What remains to be done is, among others, to identify Eve's capabilities and limitations when the Wegman-Carter authentication is used with a partially known key.

Two, long tag length implies long authentication keys, which is not favorable in QKD, since long authentication keys reduce the key growing rate of QKD protocols. The key consumption rate of authentication must be reduced. Therefore, there is an interest in designing authentication protocols consuming less key than the usual Wegman-Carter authentication.

One novel solution would be to use a combination of the Wegman-Carter authentication with a publicly known hash function. Authentication of this type consumes less key, but is not information-theoretically secure. Therefore, great care needs to be taken when using such authentication primitives in the context of QKD.

Another solution would be to authenticate through a secret (but fixed) hash function combined with a (varying) one-time-pad (OTP) key. If the OTP key is completely secret, then this type of authentication is unbreakable. If the OTP key is partially known to Eve, then she can gain some information on the secret hash function. Eve's knowledge of the secret hash function increases as the number of authentication with partially known OTP key increases; and finally Eve can gain enough knowledge about the secret hash function. This results in the security breach of the authentication. The question now would be after how many rounds Eve can gain enough information on the secret hash function; and we try to answer this question in this thesis.

This thesis is organized as follows: In Chapter 2, we briefly explain how QKD works, why authentication is important, and which type of (classical) authentication is used in QKD. Then in Chapter 3, we investigate properties of a strongly universal$_2$ hash function family, and discuss Eve's capabilities and limitations when using this family of hash functions with a partially known key. Paper A summarizes the results. In Chapter 4, we first study vulnerability of a simplified authentication protocol intended to rule out a man-in-the-middle attack on QKD, and the result is summarized in Paper B. Then we estimate the lifetime of authentication with encrypted tags which was proposed for use in QKD. The important parameters here are the length of the secret key used for authentication and Eve's partial knowledge of the encryption key. Furthermore, we perform experiments

with some family of Strongly Universal hash functions to support the theoretical estimate. Manuscript C at the end of this thesis contains these last results. In the last chapter, we draw conclusions and give further remarks about possible extensions to our work.

Part I

Classical Authentication in Quantum Cryptography

2

Quantum Cryptography and Classical Authentication

QKD is an elegant use of quantum mechanics in secure key distribution, and it is one application of quantum physics at the individual quanta level [5]. Keys generated from QKD are unconditionally secure provided that an immutable channel is used between the communicating parties. In this chapter, we explain how QKD works; why it is necessary to authenticate classical messages in QKD; and what type of (classical) authentication is used in QKD.

2.1 Quantum key distribution

We focus on the BB84 protocol [2] which consists of five steps: raw key generation, sifting, error estimate and reconciliation, privacy amplification, and authentication. Other QKD protocols also consist of these five steps, but there are variations in some of these steps as to how they are done in practical implementations.

Let us now briefly explain each step; see [13–17] for detailed explanations.

- Raw key generation: Alice sends a series of single photons each modulated in a random basis, either in rectilinear basis of vertical and horizontal, or diagonal basis of 45° and 135°, with a random value 0 or 1 to Bob. For example, in the rectilinear basis 0 is encoded as a horizontal state and 1 as a vertical state, and in the diagonal basis 0 is encoded as a 45° state and 1 as a 135° state. Bob chooses his measurement basis randomly and independently from Alice and reads the values. Then he sends Alice an authenticated time stamp to end the quantum transmission. Now they have two random bit sequences called raw keys, of which at most 75% is the same.

- Sifting: After the quantum transmission is over, Bob publicly announces his measurement basis, but *not* his measurement results, to Alice, and Alice responds to him with a message saying which bases are wrong. Then they discard all cases

where Bob chose a different basis: This is called sifting. They now have two almost identical smaller keys, that Eve perhaps has some knowledge of.

- Error reconciliation and estimation: To reconcile the two almost identical sifted keys, Alice sends error-correction information (random maps and the output values) to Bob, and error-corrects the sifted key that she shares with Bob. Bob responds by a message that signals which subsets matched and which subsets were successfully error-corrected, and also indicates the error rate of the sifted key; in simple schemes this can be used as error estimate.

- Privacy amplification: It is possible that some information is leaked to Eve during error correction. Therefore, to further increase the secrecy of the error corrected keys, Alice and Bob perform privacy amplification. This is done by Alice choosing a random map, and sending that over the classical channel, whereafter Alice and Bob apply this map to their respective reconciled keys. It is important to note in here that Eve's information on the key after privacy amplification is *not* reduced all the way to zero, but it is very small.

- Authentication: As we shall see later, it is crucial to authenticate some (or all) of the classical messages communicated during the public discussion. As to why authentication is important and how it is achieved, we will come back to these later in the following sections.

As noted above, except for the raw key generation, all the other steps are performed on the public communication channel, see Figure 2.1. This tells us how important the public channel is.

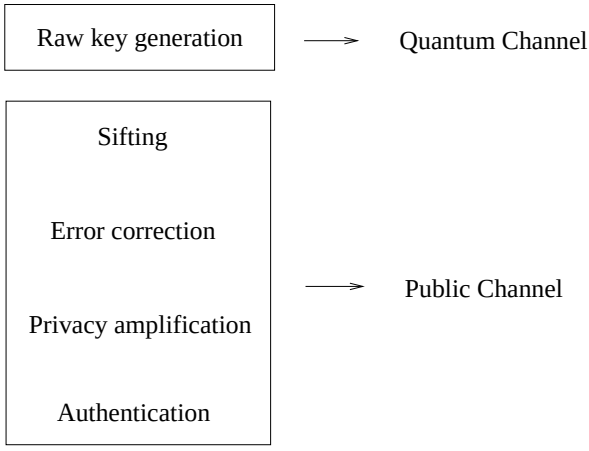


Figure 2.1: *QKD as a whole.*

2.2 The necessity of authentication in QKD

Practical implementation of QKD protocols requires an immutable public channel. In case the public channel is not immutable, the eavesdropper (Eve) can easily mount a MITM attack, since Eve can control both the quantum and the public channels. In particular, in a MITM attack on a QKD protocol, Eve first cuts the quantum and the public channels and connects them to her QKD devices; then she impersonates Bob to Alice and Alice to Bob during the quantum transmission process and the subsequent public discussions, see Figure 2.2. For the attack to be successful Eve needs, among other things, to sub-

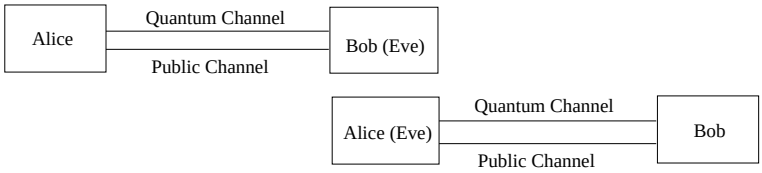


Figure 2.2: *Man-in-the-middle (MITM) attack on QKD.*

stitute the classical message from one legitimate user (Alice) to the other (Bob) without being noticed. Eve can do this without being noticed if the public channel is not authenticated. To prohibit such an attack on QKD, proper message authentication is needed. Therefore, QKD is secure only if it is combined with an unconditionally secure message authentication scheme.

As to which phases to authenticate, we refer to [17]. Next, we briefly discuss which authentication is used in QKD and how it is performed.

2.3 Classical authentication

When we talk about authentication in this thesis, it is "classical" authentication that we are referring to, as opposed to "quantum" authentication[1]. So, in our discussion, authentication refers only to classical authentication.

Authentication is an important topic in the area of cryptography. As mentioned in the previous section, "message authentication" (MA) is crucial to the overall security of a QKD system. The goal of MA is to provide the legitimate communicating parties, Alice and Bob, with a means to make sure that they are in fact communicating with each other.

To achieve MA in QKD, Alice and Bob preshare a string of secret bits long enough to authenticate the initial round. We briefly explain how authentication is done in the context of QKD: After the quantum transmission (or raw key generation) phase is completed, Alice sends her message m_{A} along with its authentication tag t_{A} generated by using the preshared key to Bob. The message here contains the settings used for encoding/decoding on the quantum channel. Upon receiving the message-tag pair $m_{\mathrm{A}} + t_{\mathrm{A}}$, Bob verifies the authenticity of m_{A} by comparing t_{A} with a tag he generated for the message using

[1]Quantum authentication is used to authenticate quantum messages using quantum error-correcting codes [18], while classical authentication is used for classical messages.

the secret key. If they are identical, then Bob can be sure, with high probability, that the message did originate from Alice; otherwise, he rejects the message. Likewise for the messages from Bob to Alice.

When the preshared secret is used up, a portion of the generated QKD keys is used to authenticate the subsequent rounds. For this reason QKD is more accurately called Quantum Key Growing.

There are two types of message authentication codes (MACs): information-theoretically secure MACs and computational complexity based MACs. Since QKD is intended to be provably unconditionally secure, it is necessary to use the first type of MACs to guarantee the unconditional security of the whole QKD system. Hence, we focus on MACs that are unconditionally secure.

Wegman-Carter authentication (WCA) [7] is the standard unconditionally secure MAC used in QKD. WCA is based on the idea of Universal hashing, which was introduced by the same authors in 1979 [6]. The idea is as follows: A secret key K is preshared by Alice and Bob which identifies a hash function f_K from a (Strongly) Universal hash function family, which we define in the next chapter. Alice sends a message m_{A} along with its tag $t_{\mathrm{A}} = f_K(m_{\mathrm{A}})$ to Bob. Upon receiving the message-tag pair $(m_{\mathrm{A}}, t_{\mathrm{A}})$, Bob verifies whether or not the message actually came from Alice by comparing $f_K(m_{\mathrm{A}})$ to t_{A}. If they are equal, then the message m_{A} is accepted as authentic: Otherwise, it is rejected.

If Eve tries to impersonate Alice and sends a forged message m_{E} to Bob, then Eve has to generate the correct tag t_{E} for m_{E} for it to be accepted as authentic. But without the knowledge of the secret key K, all tags are equally likely for m_{E}. Which means that her chance of success in this case is $1/|\mathcal{T}|$, where $|\mathcal{T}|$ is the number of all possible tags.

Eve can also try to wait until seeing a valid message-tag pair $(m_{\mathrm{A}}, t_{\mathrm{A}})$ from Alice and substitute m_{A} with her fake message m_{E}. Even in this case, if the key is unknown to Eve, the probability of t_{A} being the correct tag for Eve's message m_{E} is again exactly $1/|\mathcal{T}|$.

More on WCA in the context of QKD will be discussed in the next chapter.

3

Strongly Universal$_2$ Hash Functions

Since the introduction of universal hash functions by Carter and Wegman [6] in 1979, it has been extensively studied; and D. Stinson formalized the definitions of strongly universal$_2$ (SU$_2$) and ϵ-almost strongly universal$_2$ (ϵ-ASU$_2$) hash functions in [8]. The connection between these two different classes is that SU$_2$ hash functions are often needed as building blocks of ϵ-ASU$_2$ hash functions. It was Wegman and Carter [7] who first proposed to use ϵ-ASU$_2$ hash functions for unconditionally secure authentication purposes, hence the name Wegman-Carter authentication (WCA). This chapter is devoted to studying of properties of specific SU$_2$ hash function families.

After providing some definitions in Section 3.1, we briefly discuss the WCA in the following section. In Section 3.3 and 3.4, examples of SU$_2$ hash function classes and their properties are presented, respectively. At the end, we summarize the results in this chapter.

3.1 Definitions

To begin with, some notation is in order. For the rest of this thesis, $\mathcal{M}$ and $\mathcal{T}$ denote finite sets of messages and tags, respectively, where the size $|\mathcal{M}|$ of $\mathcal{M}$ is greater than or equal to the size $|\mathcal{T}|$ of $\mathcal{T}$. The set of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is denoted as $\mathcal{H}$.

Definition 3.1 (Universal hash functions). Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is *Universal$_2$* if there exists at most $|\mathcal{H}|/|\mathcal{T}|$ hash functions $h \in \mathcal{H}$ such that $h(m_1) = h(m_2)$ for any two distinct $m_1, m_2 \in \mathcal{M}$.

Definition 3.2 (ϵ-Almost Strongly Universal hash functions). Let $\mathcal{M}$ and $\mathcal{T}$ be as before. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is *ϵ-Almost Strongly Universal$_2$ (ϵ-ASU$_2$)* if the following two conditions are satisfied:

(a) The number of hash functions in $\mathcal{H}$ that takes an arbitrary $m_1 \in \mathcal{M}$ to an arbitrary $t_1 \in \mathcal{T}$ is exactly $|\mathcal{H}|/|\mathcal{T}|$.

(b) The fraction of those functions that also takes an arbitrary $m_2 \neq m_1$ in $\mathcal{M}$ to an arbitrary $t_2 \in \mathcal{T}$ (possibly equal to t_1) is at most ϵ.

If $\epsilon = 1/|\mathcal{T}|$, then $\mathcal{H}$ is called Strongly Universal$_2$ (SU$_2$).

Definition 3.3 (Statistical (or variational) distance). The statistical distance between two probability distributions u and v on a set, say X, denoted as $d(u, v)$, is defined as

$$d(u, v) = \frac{1}{2} \sum_{x \in X} |u(x) - v(x)|.$$

We now turn to one usage of these function classes.

3.2 Wegman-Carter Authentication

After introducing the idea of Universal hash functions in [6], Wegman and Carter presented how Universal hash functions can be applied to the construction of unconditionally secure authentication codes in [7], namely WCA. Universal hash functions can not only be used for unconditionally secure authentication, but also be used for error-correction and privacy amplifications [11, 14–16]. Here we look at their use in authentication.

As we can see from the definition, SU$_2$ hash functions can be applied to authentication in a natural way. By sharing a secret key K long enough to identify a hash function f_K from an SU$_2$ family in advance, the communicating parties, Alice and Bob, can use f_K to authenticate a message m from, say, Alice to Bob. Alice sends (m, t), where $t = f_K(m)$, to Bob. Upon receiving the message-tag pair (m, t), Bob verifies the authenticity of m by comparing $f_K(m)$ with t. If they are identical, then m is accepted as authentic. Otherwise, it is rejected.

What happens if the eavesdropper Eve tries to impersonate Alice to Bob and send m_{E} to him? What if she sees a valid massage-tag pair (m, t) and substitutes the message with her own? In the first case, Eve needs to generate the valid tag for m_{E}. If the key K is completely secret, then all tag values are equally likely for m_{E}. This means that her chance of success in this case is $1/|\mathcal{T}|$. In the second case, the probability of t being the correct tag for m_{E} when K is completely secret is again $1/|\mathcal{T}|$. In other words, she is not in an improved situation even after seeing a valid message-tag pair.

What is important to note here is that the key must be used only once, since the definition of a SU$_2$ hash function family says nothing about what happens if the same key is used twice. It may happen that two message-tag pairs reveal enough information about the secret key so that Eve can generate the valid tag for her (forged) message. This means that the key consumption rate of authentication using SU$_2$ hash functions is high, because, in most well known examples of SU$_2$ hash function families, the key length is longer than the message length. More specifically, the key length grows linearly as the message length grows. In practice, however, we want the required key length for authentication to be shorter than the message length.

By using ϵ-ASU$_2$ hash functions, where the security parameter is relaxed from $1/|\mathcal{T}|$ to $\epsilon > 1/|\mathcal{T}|$, the required key length can be reduced significantly. To be more specific, let us briefly review the Wegman-Carter construction of ϵ-ASU$_2$ hash functions. Let $\mathcal{M}$

be the set of all messages of length i, and $\mathcal{T}$ be the set of all tags of length j. Let[1] $L = j + \log\log i$. Let $\mathcal{H}$ be a set of SU_2 hash function family from the set of strings of length $2L$ to the set of strings of length L. Now let $\mathcal{H}'$ be the set of hash functions from $\mathcal{M}$ to $\mathcal{T}$ constructed as follows. A message $m \in \mathcal{M}$ is first broken into substrings of length $2L$. If needed, the last substring is padded with zeros. Thus, the message is broken into $\lceil i/2L \rceil$ substrings. Then, a hash function $h_1 \in \mathcal{H}$ is applied to all the substrings and the resulting outcomes are concatenated. The length of the concatenated strings is now roughly half the length of the original message. We repeat this process using $h_2, h_3, \cdots \in \mathcal{H}$ until only one substring of length L remains. The least significant j bits of this last substring is taken as a tag for the message. The sequence of these hash functions $(h_1, h_2, \cdots)$ form a hash function $h' \in \mathcal{H}'$. The length of these sequence of hash functions is $\log i - \log j$. The key needed to identify h' is the concatenation of the keys needed to identify $h_1, h_2, \cdots$. If the hash function family $\mathcal{H}_1$, which will be introduced in the next section, is used for $\mathcal{H}$, then the key length for $\mathcal{H}'$ will be $4L \log i$. This family of hash functions $\mathcal{H}'$ is $2/|\mathcal{T}|$-ASU_2, see [7] for details.

The above construction of ϵ-ASU_2 hash functions shows that the key length for this family increases logarithmically as the message length increases. That is why ϵ-ASU_2 hash functions are suitable for authentication in practice, especially in QKD. We note here again that ϵ-ASU_2 hash functions, however, can be constructed using SU_2 hash functions as we have seen above.

To be able to use the same hash function many times, Wegman and Carter also proposed authentication using encrypted tags in [7]. In particular, a message m is first hashed by a secret hash function f to $f(m)$, then $f(m)$ is encrypted with a one-time-pad key K to generate the tag t. The key length in this case asymptotically approaches the tag length. We study this type of authentication in detail in the next chapter.

We next present some examples of SU_2 hash function families, which are taken from the original Carter and Wegman paper [6], and study their properties.

3.3 Examples of SU_2 families

There are several SU_2 hash function families presented in Carter and Wegman [6]. We present two of them in this section. For different constructions of SU_2 hash functions, one can refer to D. Stinson [8–11], where a couple of SU_2 families, various combinatorial constructions, and the connections between error-correction codes and SU_2 hash function families are discussed.

3.3.1 The family $\mathcal{H}_1$

The first family is denoted $\mathcal{H}_1$, which was originally constructed by Carter and Wegman in [6]. Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets of size 2^i and 2^j, respectively, with $j \leq i$. Let p be the smallest prime number greater than 2^i. For each $q \in \mathbf{Z}_p \setminus \{0\}$ and $r \in \mathbf{Z}_p$, define a hash function $f_{(q,r)} : \mathcal{M} \to \mathcal{T}$ by the following rule

$$f_{(q,r)}(m) \equiv ((mq + r) \mod p) \mod |\mathcal{T}|. \tag{3.1}$$

[1] Throughout this thesis, log stands for the binary logarithm.

Then, $\mathcal{H}_1 = \{f_{(q,r)} : q \in \mathbf{Z}_p \setminus \{0\} \text{ and } r \in \mathbf{Z}_p\}$ is close to being an SU$_2$ hash function family. Close in the sense that for a randomly chosen $m \in \mathcal{M}$ there are slightly more hash functions in $\mathcal{H}_1$ that map m to small tag values than to large tag values. The required key length to identify a hash function in this family $\mathcal{H}_1$ is $\log(p(p-1))$.

We observe the following interesting property of this family of hash functions. When the message length i is chosen such that $p = 2^i + 1$ is a prime, for any choice of a hash function $f \in \mathcal{H}_1$, the uniform distribution on the set $\mathcal{M}$ induces on $\mathcal{T}$ a distribution that is close to the uniform distribution (on $\mathcal{T}$) within a statistical distance $1/|\mathcal{M}| = 2^{-i}$. This easily follows from the following proposition.

Proposition 3.1
Let $\mathcal{M}$ and $\mathcal{T}$ be as defined above. If i is chosen such that $p = 2^i + 1$ is a prime, then for any $f \in \mathcal{H}_1$ and $t \in \mathcal{T}$,

$$\frac{|\mathcal{M}|}{|\mathcal{T}|} - 1 \leq |f^{-1}(t)| \leq \frac{|\mathcal{M}|}{|\mathcal{T}|} + 1. \tag{3.2}$$

Proof: If $|\mathcal{M}| = 2^i$ and $p = 2^i + 1$ is a prime, then, for any integer $0 < q < p$ and $0 \leq r < p$,

$$\{(mq + r) \mod p : m \in \mathcal{M}\}$$

is a subset of $\mathbf{Z}_p$ of size $|\mathbf{Z}_p| - 1$, since

$$(mq + r) \mod p \equiv (m'q + r) \mod p$$

implies $m = m'$. Therefore, there are in general at most $|\mathcal{M}|/|\mathcal{T}| + 1 = 2^{i-j} + 1$ and at least $|\mathcal{M}|/|\mathcal{T}| - 1 = 2^{i-j} - 1$ elements in $\mathcal{M}$ that hash to a $t \in \mathcal{T}$ by any hash function $f \in \mathcal{H}_1$. □

Remark: In fact, for all $f \in \mathcal{H}_1$ and $t \in \mathcal{T}$,

$$|f^{-1}(t)| \in \left\{ \frac{|\mathcal{M}|}{|\mathcal{T}|} - 1, \frac{|\mathcal{M}|}{|\mathcal{T}|}, \frac{|\mathcal{M}|}{|\mathcal{T}|} + 1 \right\}. \tag{3.3}$$

This proposition implies that when $\mathcal{H}_1$ is constructed on a set $\mathcal{M}$ of messages such that $|\mathcal{M}| + 1$ is a prime, then all hash functions in the family behave equally well in the following sense. That is, for any hash function $f \in \mathcal{H}_1$, as long as the message $m \in \mathcal{M}$ is chosen according to the uniform distribution on $\mathcal{M}$, then $f(m)$ behaves as taken according to a $1/|\mathcal{M}|$-almost uniform[2] distribution on $\mathcal{T}$.

3.3.2 The family $\mathcal{H}_3$

Besides $\mathcal{H}_1$, two other SU$_2$ hash function families were proposed in Carter and Wegman [6]. One of them is denoted $\mathcal{H}_3$. Here is how it is constructed: If the elements of $\mathcal{M}$ and $\mathcal{T}$ are vectors over the field of binary numbers, then $\mathcal{H}_3$ is the set of all linear transformations from $\mathcal{M}$ to $\mathcal{T}$. More specifically, let $\mathcal{M}$ and $\mathcal{T}$ respectively be the set of i-bit and j-bit

[2]Here we call a probability distribution ϵ-uniform if its statistical distance to the uniform distribution is at most ϵ.

binary numbers. Let $\mathcal{K}$ be the set of i by j Boolean matrices whose rows are from $\mathcal{T}$. For $K \in \mathcal{K}$, let $K(k)$ be the kth row of K, and for $m \in \mathcal{M}$, let m_k be the kth bit of m. Define $\mathcal{H}_3$ to be the set of functions $f_K(m) = m_1 \odot K(1) \oplus m_2 \odot K(2) \oplus \cdots \oplus m_i \odot K(i)$, where $\odot$ and $\oplus$ are the bitwise multiplication and exclusive-or operation, respectively.

For example, let $\mathcal{M}$ be the set of 8-bit binary numbers and let $\mathcal{T}$ be the set of 4-bit binary numbers. Let the key

$$K = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}_{8\times 4} \in \mathcal{K}.$$

Then, for $m = 10011011$, $t = m_1 \odot K(1) \oplus \cdots \oplus m_8 \odot K(8) = 1001$. In this example, $|\mathcal{M}| = 2^8,\ |\mathcal{T}| = 2^4,$ and $|\mathcal{H}| = 2^{8\times 4}$.

We note that the key length needed to identify a hash function in this family is $|\mathcal{M}||\mathcal{T}|$. This long key length makes this family not suitable for authentication. But understanding properties of this hash function family is important in the study of SU_2 hash functions. Next we investigate properties of this class of hash functions.

3.4 Properties of $\mathcal{H}_3$

As noted in the previous chapter, in QKD it is possible that Eve has partial knowledge of the generated key. After the preshared key is used up for authentication in the initial QKD round, a portion of the generated key is used for authentication in the later QKD rounds. What this means is that authentication is done using probably partially known key, except for the initial QKD round. In [3], the authors studied the security of the Wegman-Carter scheme in QKD context and identified a weakness in this scheme when the authentication key is partially known to Eve. The weakness is such that the WCA becomes sensitive to the choice of message if the key is partially known.

In this section, we study properties of $\mathcal{H}_3$ to exploit the above mentioned weakness in the case when this family is used for authentication with a partially known key. As previously mentioned, this family itself is not appropriate for authentication because of the long key length required, which is common to all SU_2 families. Suitable families of hash functions for authentication purposes, especially in QKD, are ϵ-ASU_2 families, which are constructed using SU_2 hash functions [7], since they consume less key than SU_2 families at the cost of increasing the security parameter from $1/|\mathcal{T}|$ to $2/|\mathcal{T}|$. The results in this section are summarized in Paper A.

When the key is completely secret, there are two possibilities for Eve to attack the system. The first is to guess the tag value for her message m_E randomly, while the other is to wait for the message-tag pair (m_A, t_A). The message-tag pair will give her some information on the key. But in both cases her chances of success are the same according

to the definition of SU$_2$, and they are

$$P(T_E = t) = \frac{1}{|\mathcal{T}|}, \tag{3.4}$$

and

$$P\left(T_E = t \mid h_K(m_A) = t_A\right) = \frac{1}{|\mathcal{T}|}. \tag{3.5}$$

In practice, information leakage in the quantum channel is unavoidable. Eve's knowledge can be reduced significantly by privacy amplification but not all the way to zero. Hence, it is essential to assume that Eve always has partial knowledge of the key generated by the previous QKD rounds.

If Eve uses all her knowledge to eliminate some keys, then denoting the remaining set of keys as $\mathcal{H}_E$ she will have

$$\mathcal{H}_E = \mathcal{H} \setminus \{h_1, \ldots, h_n\}. \tag{3.6}$$

Let $s = |\mathcal{H}_E|/|\mathcal{H}_A|$. Then, from Eve's perspective the true key is drawn from the remaining $|\mathcal{H}_E| = s|\mathcal{H}|$ keys with equal probability. Therefore,

$$P(T_E = t) \leq \sum_{1}^{|\mathcal{H}|/|\mathcal{T}|} \frac{1}{s|\mathcal{H}|} = \frac{1}{s|\mathcal{T}|}. \tag{3.7}$$

Now, when Eve picks up a message-tag pair, she again gains additional information that increases her knowledge about the key. The message-tag pair (m_A, t_A) that Eve receives from Alice identifies a subset of keys (hash functions) of size $|\mathcal{H}|/|\mathcal{T}|$ from which the key must have been drawn:

$$\mathcal{H}_A = \{h \in \mathcal{H} : h(m_A) = t_A\}. \tag{3.8}$$

The final set of possible keys now is not $\mathcal{H}_A$ but $\mathcal{H}_{AE} = \mathcal{H}_A \cap \mathcal{H}_E$. For a SU$_2$ hash function family, when

$$|\mathcal{H}_{AE}| \leq \frac{|\mathcal{H}|}{|\mathcal{T}|^2}, \tag{3.9}$$

there may exist messages m that are such that

$$\forall h_1, h_2 \in \mathcal{H}_{AE},\ h_1(m) = h_2(m). \tag{3.10}$$

That is, for this message, all remaining keys map to the same tag. The number of messages with this property will increase as $|\mathcal{H}_{AE}|$ decreases from $|\mathcal{H}|/|\mathcal{T}|^2$. For the family $\mathcal{H}_3$, this happens when Eve has complete knowledge of at least one row of the key. Since this analysis is mainly focused on the worst case scenario, we restrict ourselves to this case. In this case, the number of messages she can generate the correct tag for when she has seen the message-tag pair (m_A, t_A) is twice as large as before she has seen the message-tag pair. More generally, when Eve has knowledge of nj bits of the ij-bit key, the number of messages that she can generate the correct tag for by any of the remaining keys in $\mathcal{H}_{AE}$ is at most $2^{n+1} - 2$. While the number of such messages is at most $2^n - 1$ when she does not know $\mathcal{H}_A$.

Another method important in QKD is to influence Alice's message so that Eve can create the correct tag t_E for her message m_E. This is possible, because Eve can influence the content of Alice's message by accessing and changing what happens on the quantum channel.

Suppose that Eve has complete knowledge on, say, two rows. That means she knows $2i$ bits of the ij-bit key, where i and j are the bit-length of the message and tag, respectively. Assume, without loss of generality, that Eve has perfect knowledge of the first and second row of the key K. Assume also that Eve has message m_E whose first and second bit values are zeros, say $m_E = \underbrace{00101\cdots01}_{i}$. Then, since

$$t_E = m_E(1) \odot K(1) \oplus m_E(2) \odot K(2) \oplus \cdots \oplus m_E(i) \odot K(i),$$

Eve can influence Alice's message m_A so that it is the same as m_E except at the first and second positions. Then, the message-tag pair (m_A, t_A) will give her the information she needs to create the correct tag t_E. In this example, if $m_A = 1\ 0\ m_E(3)\ \cdots\ m_E(i)$, she just needs to calculate $K(1)\ \oplus\ t_A$, and likewise for the other cases.

Therefore, when Eve has knowledge of nj bits of the ij-bit key, she needs to influence at least $i-n$ bits of Alice's i-bit message in order to be able to create the correct tag for her message. This is, however, a serious restriction for Eve, because she needs to influence a large portion of Alice's message. This is due to the very long key length required by $\mathcal{H}_3$, see the discussion above.

3.5 Summary

In this chapter, we first presented definitions of Universal hash functions, and then discussed their use in unconditionally secure authentication. Then we presented two hash function families, namely $\mathcal{H}_1$ and $\mathcal{H}_3$, which are taken from the original Carter and Wegman paper [6], and studied the properties of these classes of hash functions.

Regarding the family $\mathcal{H}_1$, we observed an important property of each individual hash function when this family is constructed on a set $\mathcal{M}$ of messages such that $p = |\mathcal{M}| + 1$ is a prime. In this case, all hash functions in $\mathcal{H}_1$ behave equally well in the following sense. That is, for any hash function $f \in \mathcal{H}_1$, the uniform distribution on $\mathcal{M}$ induces a $1/|\mathcal{M}|$-almost uniform distribution on $\mathcal{T}$.

For the family $\mathcal{H}_3$, we have studied and identified Eve's possibilities when her partial knowledge of the secret key is such that (3.9) is satisfied. There are messages for which she can generate the correct tag for. This happens when she has complete knowledge of a row of the secret key. In this case, seeing a valid message-tag pair enables Eve to generate the correct tag for twice as many messages as before seeing a message-tag pair. Eve can also influence Alice's message by influencing what happens on the quantum channel so that the message-tag pair from Alice will give her enough information to create the valid tag for her forged message. This, however, is very restrictive and difficult for Eve to achieve, since she needs to influence a large portion of Alice's message.

4

Security Analysis of Authentication with Reduced Key Consumption

When using an ϵ-ASU$_2$ hash function family for unconditionally secure message authentication, the key length is shorter than the message length when the message is long. In Wegman and Carter [7], for instance, to authenticate an i-bit message with a j-bit tag the required key length is equal to $4(j + \log\log i)\log i$, see Section 3.2. We refer to M. Atici and D. Stinson [12] for other constructions. For short messages, however, the required key length is longer than the message length. This is a problem in QKD where the messages to be authenticated at times are short [19]. That affects the key growth rate of QKD, since a portion of the generated key is reserved for subsequent authentications. Therefore, it is necessary to reduce the key consumption rate of the authentication system for short messages in order to improve the key growing rate.

This chapter is focused on the security analysis of two types of authentication methods aiming at reducing the key consumption rate. In the first half of this chapter, we study vulnerabilities of a novel authentication algorithm ruling out a man-in-the-middle (MITM) attack in QKD proposed by M. Peev *et el.* in [19]. Paper B presents the results on this. The remainder of this chapter is an overview of Paper C, where we study the lifetime of authentication using encrypted tags, which is is unconditionally secure only if the authentication key is completely secret, under the assumption that the key is partially known.

4.1 A novel authentication protocol

In [19], the authors propose an authentication primitive which aims at decreasing the key consumption for the authentication purposes in QKD, and in turn to improve the efficiency of the key growth in QKD. The algorithm works as follows. Let $\mathcal{M}$ be the set of all binary strings of length m (or the set of all messages of length m), and let $\mathcal{T}$ be the set of all binary strings of length n with $n < m$ (or the set of all tags of length n). A message m_{A} is first mapped from $\mathcal{M}$ to $\mathcal{Z}$, where $\mathcal{Z}$ is the set of all binary strings of length r

with $n < r < m$, by a single publicly known hash function f so that $z_A = f(m_A)$. And then, z_A is mapped by a secret $h_k \in \mathcal{H}_{\mathcal{Z}}$ to a tag $t_A = h_k(z_A)$, where $\mathcal{H}_{\mathcal{Z}} : \mathcal{Z} \mapsto \mathcal{T}$ is a Strongly Universal$_2$ (SU$_2$) family of hash functions [6] and the subscript k is the secret key needed to identify a hash function. The message-tag pair $m_A + t_A$ will be sent over the public channel. To authenticate the message $m_A \in \mathcal{M}$, the legitimate receiver computes $h_k(f(m_A))$ and compares it to t_A. If they are identical then the message will be accepted as authentic, otherwise it will be rejected. Since r is fixed independently of m, the key length required for authentication is constant regardless of the message length to be authenticated.

This authentication algorithm is claimed [19] to be secure with a probability ϵ of Eve being able to create the correct tag for her fake message. In [19], this is calculated as[1]

$$\epsilon = \epsilon_1 + \epsilon_2, \tag{4.1}$$

where $\epsilon_2 = 1/|\mathcal{T}|$ which is the probability of guessing the correct tag when a SU$_2$ hash function family is used and ϵ_1 is the probability that the message m_A and Eve's modified message $m_E(\neq m_A)$ yield the same value under the publicly known hash function f.

4.1.1 The problem

Whenever $f(m_E) = f(m_A)$, that is Eve's message collides with Alice's message under f, Eve can just send $m_E + t_A$, since $t_E = t_A$. In the QKD context, m_A contains the settings used for encoding/decoding on the quantum channel, error-correction information and description of a random map depending on the phase when it is sent.

In a full MITM attack on a QKD protocol, Eve impersonates Bob to Alice and Alice to Bob during the quantum transmission process and the subsequent public discussions.

In [19], security is derived under the explicit assumption that Eve has a fixed message. In this special case, the result holds, but in generic QKD, Eve is *not* restricted to one message m_E.

We consider BB84 [2] with simple reconciliation and privacy amplification; and *immediate authentication* of each phase as our first example. This would consist of, in order, raw key generation; sifting and immediate authentication; one-way error correction and immediate authentication; one-way privacy amplification and authentication (see, e.g., [21] Chapter 12).

Eve receives and measures the qubits that Alice has sent to Bob, in her choice of basis. We note here that although QKD requires that Bob randomly selects the basis to measure the qubits in, Eve can ignore this requirement. At the same time she chooses a set of qubits in, again, not necessarily random states and sends these to Bob. After Bob receives and measures the qubits sent by Eve in a randomly selected basis, he sends an authenticated time stamp to Alice to end the quantum transmission phase.

Now Alice sends $m_A + t_A$, where m_A contains the settings used for encoding/decoding on the quantum channel, to Bob. Eve intercepts $m_A + t_A$ and calculates $f(m_A)$ and compares it with $f(m_E)$. If $f(m_E) = f(m_A)$, Eve can just send $m_E + t_A$ to Bob. Otherwise, Eve can search for a message m'_E with $d_{\text{Hamming}}(m_E, m'_E) = 1$ (or "small") such that $f(m'_E) = f(m_A)$. In other words, she tries to find a collision between m_A and m'_E under

[1] Actually, $\epsilon \leq \epsilon_1 + \epsilon_2$; eqn. (4.1) is an upper bound rather than an equality.

f such that m'_{E} is close to m_{E}, and it is well known that such collisions may exist for many hash functions and in fact do exist for well-known examples [22, 23]. Eve can now send the message-tag pair $m'_{\mathrm{E}} + t_{\mathrm{A}}$ knowing that Bob will accept the message m'_{E} as authentic.

Searching for a collision requires Eve to have sufficient computing power, but usually in QKD no bounds are assumed on Eve's computing power. One should also note that the computing power needed may be lower than one would first expect [22, 23]. Even without sufficient computing power, however, Eve can make a list of different values of m'_{E} and the corresponding value of $z'_{\mathrm{E}} = f(m'_{\mathrm{E}}) \in \mathcal{Z}$ in advance, and save it in her device. With a pre-chosen m_{E}, a list of pairs $(m'_{\mathrm{E}}, z'_{\mathrm{E}})$ and her received $m_{\mathrm{A}} + t_{\mathrm{A}}$, Eve can just compute $z_{\mathrm{A}} = f(m_{\mathrm{A}})$ and pick m'_{E} from her list corresponding to z_{A}, and then send $m'_{\mathrm{E}} + t_{\mathrm{A}}$. She can even make a partial list, and simply wait for the first match to occur. If she is able to make a full list (one message m'_{E} for each possible z_{A}), or has sufficient computing power, she is certain of success in the sifting phase every time she performs the MITM attack.

The remaining steps are completed by sending random parity maps over the classical channel, and in case of error correction also the parity values [13–16]. In the case of error correction, Eve intercepts the authenticated error-correction information sent by Alice to Bob, and error-corrects the sifted key that she shares with Alice. She then searches for *non-random* maps and corresponding output of the sifted key shared with Bob, that makes her message collide with Alice's under f. She sends the resulting message to Bob along with Alice's tag, which will then be accepted by Bob. Bob responds by an authenticated message that signals which subsets matched and which subsets were successfully error-corrected, and also indicates the error rate of the sifted key; in this simple scheme this is used as error estimate. Eve modifies her corresponding but still waiting response to Alice so that it will collide with Bob's message under f.

The privacy amplification is performed by Alice choosing a random map, and sending that over the classical channel, whereafter Alice and Bob apply this map to their respective reconciled keys. Here, Eve intercepts the description of the map and the tag, and privacy amplifies the reconciled key (shared with Alice) using the received map. She then searches for a new *non-random* map to use for privacy amplification with Bob that makes the message coincide with Alice's under f. Then, Eve sends the chosen map along with Alice's tag to Bob, who will accept them and privacy amplify his error-corrected key accordingly.

4.1.2 Countermeasures

The situation is improved if postponed authentication is used, that is, the messages are sent in each phase as usual (sifting, error correction and privacy amplification, etc.) but not authenticated until the end of the round. In this case, Eve's freedom to change her message is restricted to the message part in the last phase. And this severely restricts Eve's possibilities, even though an attack is still possible as is shown in [20].

Another more effective improvement is to use secret key in an additional phase of the protocol [17]. Another suggestion is to one-time pad the reconciliation procedure [24].

4.1.3 Summary

This brief review shows that the proposed method is insecure when used in a generic QKD protocol. The main problem is that Eve is not limited to a fixed (random) message, but can in fact choose what message to send, and can check if her chosen message gives the same tag as Alice's message, since the first-step hash function f is publicly known.

Using extra shared secret key for an extra authentication in one of the phases probably improves the situation, but it should be stressed that, unlike Wegman-Carter authentication, the security of the proposed authentication procedure is highly dependent of the context in which the authentication is applied.

4.2 Authentication using encrypted tags

Authentication using encrypted tags is of particular interest in QKD because of the reduced key consumption rate of authentication. In this section, we estimate the lifetime of this type of authentication, where the encryption is XORing with a one-time-pad (OTP), in the case when the OTP is partially known.

Authentication of this type works in the context of QKD as follows: The legitimate communicating parties, Alice and Bob, share a secret but fixed hash function f taken at random from an SU_2 hash function family and a short secret key to be used as OTP in advance. During the public discussion phase of each QKD round, Alice sends the classical message and tag pair $m + t$ with $t = f(m) \oplus K$, where K is an OTP, to Bob. In the initial QKD round, K is the preshared secret key. If everything goes well and a string of keys are successfully generated in the initial QKD round, then a portion of this newly generated key is used as the OTP key for authentication in the subsequent QKD round. Upon receiving the message-tag pair (m, t), Bob verifies if the message m did originate from Alice by comparing $f(m) \oplus K$ to t: If they are identical, then he accepts m as authentic, otherwise, he rejects it.

This authentication was also mentioned in Cederlöf [4], where the author briefly discussed that this type of authentication would not be advisable for use in an environment where some partial information on the OTP key are leaked to the eavesdropper Eve. Because partial knowledge of the OTP key K, along with m and $f(m) \oplus K$, help Eve gain information on $f(m)$, which gives her partial knowledge of the secret hash function f. And when the number of authentications with a partially known K increases, probably Eve's knowledge of f also increases.

Information leakage is unavoidable in QKD. Eve may have some partial knowledge of the generated key, a portion of which is used as the OTP for later authentication. Hence, the OTP is probably partially known. For this reason, we study the lifetime of this authentication in the case when the OTP is partially known in each round.

In the case when the OTP key K is completely secret and Eve's goal is to be able to create a valid tag t_{E} for her message m_{E}, the best attack for Eve would be to guess the value of t_{E}. Since all tag values are possible, the probability of each guess succeeding is $1/|\mathcal{T}| = 2^{-\log|\mathcal{T}|}$, which implies that the expected lifetime

$$n = |\mathcal{T}| = 2^{\log|\mathcal{T}|} \tag{4.2}$$

is exponential in the tag length $\log|\mathcal{T}|$. Furthermore, she can gain no knowledge about the secret hash function f from guessing, because K in the current round is independently distributed from previous rounds.

We now study how this exponential lifetime behavior would change if Eve has some knowledge of K in each round. In particular, we estimate the lifetime until Eve gains complete knowledge of the secret hash function f (taken at random from an SU_2 family), under the assumption that she has a fixed amount of partial knowledge of K in each round. We note that the lifetime until f is found and the lifetime until Eve gains the information she needs to generate the correct tag for her message is different. Eve may be able to generate the correct tag for her message even when the number of remaining hash functions is high. However, we estimate the lifetime until f is found.

Notation. In what follows, $\mathcal{H}$ is an SU_2 hash function family with $|\mathcal{H}| = H + 1$, and $\mathcal{H}_i$, for $i = 1, 2, \cdots$, are subsets of $\mathcal{H}$, unless we explicitly state that $\mathcal{H}_1$ is the family of hash functions introduced in the previous chapter. At each round, say the i-th round, Eve can identify a set $\mathcal{H}_i$, which consists of the secret hash function and a number h of false matches, based on her partial knowledge of the OTP key K. Therefore, we view Eve's information on the key as $-\log(h/H)$. The number of false matches in the intersection $\cap_{j=1}^{i}\mathcal{H}_j$ is denoted as a random variable X_i. Lifetime and expected lifetime are denoted as N and n, respectively; and n_k denotes the expected lifetime when there are currently k (false) hash functions.

4.2.1 Lifetime

In each QKD round, Eve intercepts a valid (classical) message-tag pair $m + t$, where $t = f(m) \oplus K$, from, say, Alice to Bob. Eve uses her partial knowledge of K to identify possible candidates for $f(m)$. This means that in each run, Eve can identify a subset $\mathcal{H}_i$ out of all the possible hash functions in $\mathcal{H}$ by eliminating the hash functions (in $\mathcal{H}$) that do not hash m to the set of possible candidates for $f(m)$. The set $\mathcal{H}_i$ will consist of the true match (the fixed secret hash function) and a number h of false matches. Similarly, the set $\mathcal{H}$ consists of the true match and H false matches. The number of i runs will decrease the set of possible hash functions to $\cap_{j=1}^{i}\mathcal{H}_j$. In general, the remaining number of false matches in this intersection is a random variable $X_i = |\cap_{j=1}^{i}\mathcal{H}_j| - 1$. As a simplification we now assume that each trial is independent of the former, i.e., that the probability of drawing a hash function present in $\cap_{j=1}^{i-1}\mathcal{H}_j$ in run i only depends on X_{i-1}. We are interested in the expected lifetime of the system, that is, the expectation of the (random) index N that is the earliest that gives $X_N = 0$ (such that $X_{N-1} \geq 1$).

The simplest case is when $X_i = X_{i-1}h/H$, when each subset is exactly evenly distributed within the previous subsets. This is an oversimplification, but analyzing this will help in what follows. One problem is that the X_i are discrete (integer-valued) random variables; for the moment we will assume that they are continuous. In this case, if $X_0 = k$ we have $X_1 = kh/H$, $X_2 = k(h/H)^2, \ldots, X_l = k(h/H)^l$. Now, our previous demand $(X_N = 0) \cap (X_{N-1} \geq 1)$ translates into $(X_N < 1) \cap (X_{N-1} \geq 1)$, which in turn implies that $N|(X_0 = k)$ is not random in this case, but is in fact equal to n_k where

$$k(\tfrac{h}{H})^{n_k} < 1 \leq k(\tfrac{h}{H})^{n_k-1} \tag{4.3}$$

which after some algebra simplifies to

$$n_k - 1 \leq \frac{\log k}{-\log \frac{h}{H}} < n_k, \tag{4.4}$$

that is,

$$n_k = \left\lceil \frac{\log k}{-\log \frac{h}{H}} \right\rceil. \tag{4.5}$$

In particular, $n_H = \lceil \log H/(-\log(h/H)) \rceil$, which means that the lifetime of the system would be directly proportional to the key length[2] divided by the information on the OTP used in each step. This is what we would expect of a system in which there is a constant gain of information in each run.

Our goal is to show that the full system has similar behavior. There are three complicating factors: first, the random variables X_i, $i = 1, 2, \cdots$, have nonzero variance, second, the random variables are discrete while small values of k imply $\lceil kh/H \rceil = k$, and third, each trial is not independent of the former as opposed to our previous assumption.

To get closer to the real situation, we assume that $\mathcal{H}_i$ is randomly drawn without replacement from $\mathcal{H}$, where there are two types of elements: those in $\cap_{j=1}^{i-1}\mathcal{H}_j$ (X_{i-1} of them), and those outside the set. In other words, the number of hash functions in $\cap_{j=1}^{i}\mathcal{H}_j$ given X_{i-1} is hypergeometrically distributed, more specifically

$$X_i|(X_{i-1} = k) \sim \text{Hyp}\left(H, k, \frac{h}{H}\right). \tag{4.6}$$

In terms of probabilities this is

$$p_{jk} := P(X_i = j|X_{i-1} = k) = \frac{\binom{k}{j}\binom{H-k}{h-j}}{\binom{H}{h}}. \tag{4.7}$$

The expectation and variance are

$$E(X_i|X_{i-1} = k) = k\tfrac{h}{H} \tag{4.8}$$

and

$$V(X_i|X_{i-1} = k) = h\frac{k}{H}\Big(1 - \frac{k}{H}\Big)\frac{H-h}{H-1} \leq k\frac{h}{H}\Big(1 - \frac{h}{H}\Big), \tag{4.9}$$

where we have used

$$\frac{H-k}{H-1} = 1 - \frac{k-1}{H-1} \leq 1. \tag{4.10}$$

Because of the nonzero variance, the X_i will now differ from the mean value in (4.8), and our question now is if this increases the expected lifetime, and if so, how much.

The expected lifetime time when k (false) hash functions remain is (cf. above)

$$n_k = E(N|X_0 = k). \tag{4.11}$$

[2]Here, the length of the key identifying the secret hash function is actually $\log(H + 1)$.

Then,

$$n_0 = 0 \tag{4.12}$$

and

$$\begin{aligned} n_k &= \sum_{j=0}^{k} E(N|X_1 = j)P(X_1 = j|X_0 = k) \\ &= \sum_{j=0}^{k} \Big(E(N|X_0 = j) + 1\Big)P(X_1 = j|X_0 = k) = 1 + \sum_{j=0}^{k} p_{jk} n_j. \end{aligned} \tag{4.13}$$

Solving for n_k gives

$$n_k = \frac{1 + \sum_{j=0}^{k-1} p_{jk} n_j}{1 - p_{kk}}, \tag{4.14}$$

and since p_{jk}, $j = 0, 1, \cdots, k$, are given explicitly above, the n_k can be calculated explicitly from this equation. For example,

$$n_1 = \frac{1}{1 - p_{11}} = \frac{1}{1 - \frac{h}{H}}. \tag{4.15}$$

This depends only on the knowledge of the key, not on the size of $\mathcal{H}$.

We want to prove logarithmic dependence of n_k on k as in (4.5) in general. By splitting the sum in (4.13) we obtain

$$n_k = 1 + \sum_{j=0}^{l} p_{jk} n_j + \sum_{j=l+1}^{k} p_{jk} n_j \leq 1 + \sum_{j=0}^{l} p_{jk} n_l + \sum_{j=l+1}^{k} p_{jk} n_k. \tag{4.16}$$

And now solving for n_k gives

$$n_k \leq \frac{1}{1 - \sum_{j=l+1}^{k} p_{jk}} + n_l, \tag{4.17}$$

where $\sum_{j=l+1}^{k} p_{jk}$ can be written as $P(X_{i-1} \geq l+1 | X_i = k)$. If $l = kh/H$ and the sum in the denominator is 0, then we have $n_k \leq 1 + n_{kh/H}$, which is exactly the logarithmic behavior we desire, since kh/H is much smaller than k when k is large.

By using the one-sided Chebyshev inequality and induction, see Paper C for details, we arrive at

$$n_k \leq \frac{1}{1 - \frac{h}{H}} + \left(1 + \frac{\frac{h}{H}}{(1 - \frac{h}{H})(k-1)}\right) \frac{\log k}{-\log \frac{h}{H}}, \quad \text{for } k > 1. \tag{4.18}$$

What can be seen from the above inequality is that if this authentication is used with a secret hash function taken randomly from an SU_2 family and a partially known OTP, then the lifetime is linear with respect to the length of the key identifying the secret hash function.

We note here again that our estimate is based on the assumption that $X_i | X_{i-1} = k$ is a hypergeometrically distributed random variable. Also, the estimate above is very loose for k small. So when k is small, the probabilities p_{jk}, $j = 0, 1, \cdots, k$, must be used to solve (4.14) for the lifetime.

Next, we present simulation results for the family $\mathcal{H}_1$.

4.2.2 Simulations for the Family $\mathcal{H}_1$

Now, we present experimental results on the lifetime of the authentication with the secret hash function f taken at random from the family $\mathcal{H}_1$ and a partially known OTP. Note that the goal is to find the secret hash function f.

Our experimental setup is as follows. We set Eve's partial information on the OTP key K to 10%. We fix $\mathcal{T}$ as the set of all 7-bit tags, and $\mathcal{M}$ varies from the set of all messages of length 9-bit through the set of all messages of length 13-bit. For each pair of $\mathcal{M}$ and $\mathcal{T}$, there is a corresponding hash function family $\mathcal{H}_1$. For a message m, the 10% information on the OTP corresponds to 10% information on $f(m)$. This implies that, at round i, we can identify a subset $\mathcal{T}_i$ of impossible outputs for $f(m)$ from $\mathcal{T}$.

At the first round, we eliminate the hash functions that map Alice's message into $\mathcal{T}_1$. At the next round, we further eliminate some hash functions from the set of hash functions remained in the first round by the same technique. This continues until there is exactly one hash function–the secret hash function–left. We repeat this process as many times as needed to reduce the standard deviation of the average lifetime to 1% of the mean. Figure 4.1 presents the obtained lifetime results. From the figure it can be seen that the

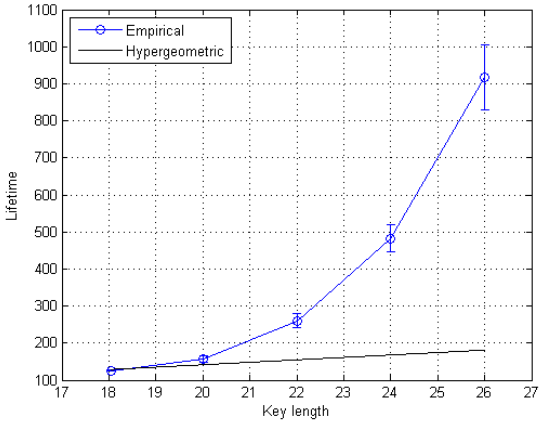
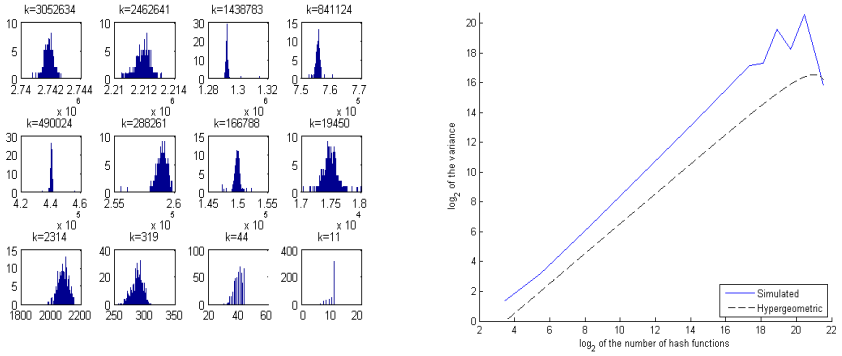


Figure 4.1: *The lifetime until the secret hash function f is found when it is taken at random from the family $\mathcal{H}_1$.*

lifetime is not as was estimated in (4.18). The lifetime exponentially increases as the length of the key increases.

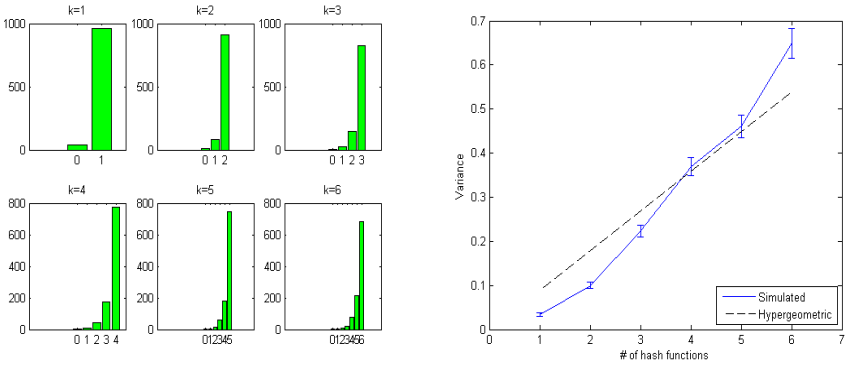
Let us recall that we obtained the estimate in (4.18) using the one-sided Chebyshev inequality, which uses the variance of $X_i|X_{i-1} = k$, see Paper C for details. Hence, the reason for the unexpected results in Figure 4.1 might be because the variance of the random variables $X_i|X_{i-1} = k$ is greater than the hypergeometric variance given in (4.9). The larger the variance of $X_i|X_{i-1} = k$ is, the bigger the lifetime is. So we simulate the variance of $X_i|X_{i-1} = k$ and compare it with the hypergeometric variance. The setup in this case is as follows: $|\mathcal{M}| = 2^{11}$, $|\mathcal{T}| = 2^7$ and the information (on the OTP) in percentage is 10%. The experiment is first run until a specified number of hash functions



(a) *Histogram.* ***(b)*** *Variance.*

Figure 4.2: *The Histogram and variance of* $X_i|X_{i-1} = k$. *The variance is plotted in* log-log *scale.*

left. Then the remaining hash functions are fixed and we look at the next round 500 times to see how many hash functions would still remain. The simulated results are as displayed in Figure 4.2.



(a) *Histogram* ***(b)*** *Variance*

Figure 4.3: *The histogram and variance of* $X_i|X_{i-1} = k$ *when* $k = 1, 2, \cdots, 6$.

Figure 4.2 tells us that if we denote by V_{sim} and V_{hyp} the simulated and hypergeometric variances, respectively, then $V_{sim} = \mathcal{O}(V_{hyp})$. Since in the figures, the solid lines represent $\log(V_{sim})$ and the dashed lines $\log(V_{hyp})$, and $\log(V_{sim}) \leq \log(V_{hyp}) + C$, for some small constant C, we have $V_{sim} \leq 2^C V_{hyp}$. Note that in the above experiments we have looked at until the case when $k = 11$. If we go further and check for the k is

small case, say, $k = 1, 2, \cdots, 6$, then we obtain the results in Figure 4.3 and Table 4.1. Remember that k is the number of false matches.

Table 4.1: *Empirical and hypergeometric mean, $E(X_i|X_{i-1} = k)$, for small k.*

k	$E(X_i\|X_{i-1} = k)$	
	simulated	hypergeometric
1	0.9633± 0.0045	0.9
2	1.8996± 0.0075	1.8
3	2.7947± 0.0086	2.7
4	3.7016± 0.0105	3.6
5	4.6465± 0.0126	4.5
6	5.5350± 0.0136	5.4

The above empirical results (in Figure 4.2 and 4.3 and Table 4.1) do not fully account for the exponential lifetime (in key length) of the system we have in Figure 4.1. This is because we have not seen any significant difference between the two variances. Therefore, we further investigate the reason for the unexpected results.

Let us take a look at the expected lifetime n_k when $k = 1, 2, \cdots, 6$. With the same experimental setup as for the simulation of the variance, we get the results in Figure 4.4. Figure 4.4 shows that the (experimental) lifetime (until f is found), for small values of k,

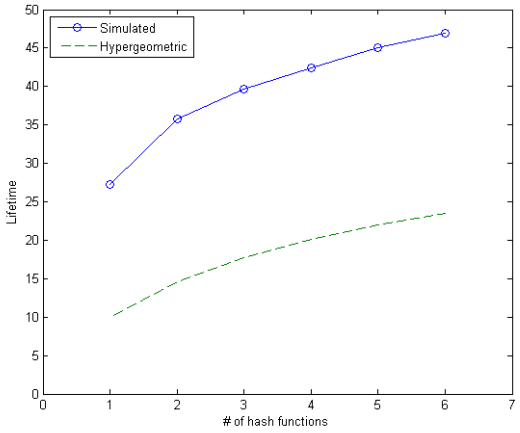


Figure 4.4: *Lifetime until f is found for small k.*

is already roughly 2 to 3 times higher than the hypergeometric lifetime. Plus, the difference between the two lifetimes is almost steady. A close inspection of the experimental data reveals that the cause for the high lifetime for small k is because of the high probabilities p_{kk}, $k = 1, 2, \cdots, 6$, see Table 4.2. Note that we can solve (4.14) for the exact lifetime once we know the probabilities p_{jk}, $j = 0, 1, \cdots, k$ and $k = 1, 2, \cdots$.

Table 4.2: *Simulated and hypergeometric probabilities* p_{jk}, $j = 0, 1, \cdots, k$ *and* $k = 1, 2, \cdots, 6$.

j	p_{jk}	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
0	Simulated	0.0367	0.0076	0.0030	0.0017	0.0004	0.0003
	Hypergeometric	0.1000	0.0103	0.0011	0.0001	0.0000	0.0000
1	Simulated	0.9633	0.0851	0.0246	0.0102	0.0044	0.0021
	Hypergeometric	0.9000	0.1825	0.0278	0.0037	0.0005	0.0001
2	Simulated	0	0.9073	0.1470	0.0445	0.0117	0.0086
	Hypergeometric	0	0.8072	0.2459	0.0500	0.0084	0.0013
3	Simulated	0	0	0.8254	0.1718	0.0586	0.0193
	Hypergeometric	0	0	0.7252	0.2946	0.0748	0.0152
4	Simulated	0	0	0	0.7717	0.1817	0.0745
	Hypergeometric	0	0	0	0.6516	0.3309	0.1008
5	Simulated	0	0	0	0	0.7432	0.2115
	Hypergeometric	0	0	0	0	0.5854	0.3567
6	Simulated	0	0	0	0	0	0.6837
	Hypergeometric	0	0	0	0	0	0.5259

The results in Figure 4.4 and 4.2 give us empirical evidence that when k is small, the lifetime (until f is found) of the system is higher than the hypergeometric lifetime. Hence, the lifetime (of finding the secret hash function) of the system is exponential in the key length, see Figure 4.1.

Now, to see whether our hypergeometric assumption is reasonable at least for certain values of k until $0 < j < k$ hash functions left, we investigate a different scenario where Eve's goal is different from finding the secret hash function f. More specifically, we experimentally observe what would happen if Eve's objective is to be able to generate the correct tag for her (forged) message, which would also lead to the security breach of the authentication. We emphasize here that our focus has been on finding f. This is different from what we are about to do, because Eve may be able to generate the valid tag for her message even when there are many remaining hash functions.

For the same experimental setup as for the previous lifetime experiment, if we check for when Eve can gain the information she needs to generate the valid tag for her message, then we obtain the results in Figure 4.5. We note that Eve can generate the correct tag for her message only if all the remaining hash functions not only map her message to the same value (say, t_{E}) , but also map Alice's message to the same value (say, t_{A} which may be different from t_{E}). Since when Alice's message hashes to the same output by all the remaining hash functions, Eve can identify the OTP. Thus, Eve can generate the correct tag for her message if it hashes to the same value by all the remaining hash functions as well.

As can be seen from Figure 4.5, the system now has a lifetime linear in the key length when Eve's objective is changed from finding f to be able to generate the correct tag for

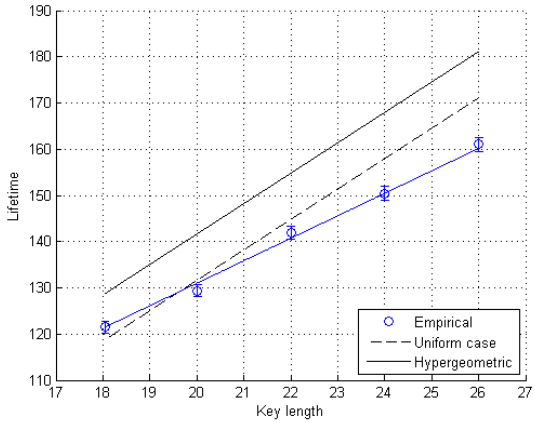


Figure 4.5: *The lifetime until f is found under the uniform and hypergeometric assumptions, and the lifetime until Eve gains enough information to generate the valid tag for her forged message.*

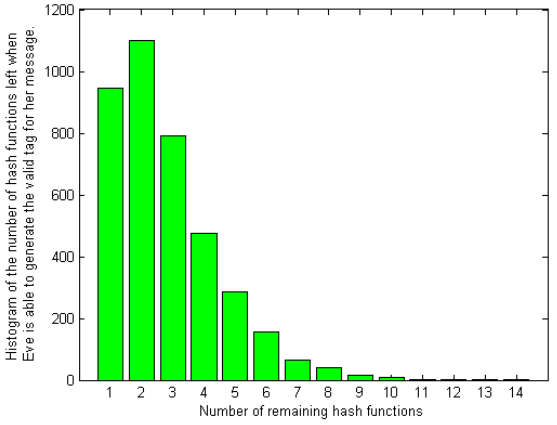


Figure 4.6: *When we look at a set of lifetime experiments, where each experiment is run until Eve is able to generate the valid tag for her message, Eve can do so when the number of remaining hash functions are just a few. The parameters are $|\mathcal{M}| = 2^{11}$, $|\mathcal{T}| = 2^7$ and $h/H = 0.9$, and in total there are 3894 experiments.*

her message. If we look carefully at the experimental data in this case, then we observe that Eve can generate the valid tag for her message when there are very few hash functions left. For instance, for $|\mathcal{M}| = 2^{11}$ and $|\mathcal{T}| = 2^7$ and $h/H = 0.9$, Eve is capable of generating the valid tag for her message mostly when the remaining number of hash

functions is less than 6, see Figure 4.6.

4.2.3 Summary

The authentication primitive that we studied in this section is not unconditionally secure if used in an environment where the one-time-pad is partially known in each round. Since information leakage in QKD is unavoidable, Eve may have partial information on the generated QKD keys, a portion of which is used as the one-time-pad in each authentication. Hence, the one-time-pad may be partially known to Eve. Eve can use her partial knowledge of the one-time-pad to gain information on the secret hash function f. As the number of authentication rounds with partially known one-time-pad increases, Eve's information on f also increases. At the end, Eve gains complete information on f, which results in the breakdown of the authentication.

We have estimated under the hypergeotmetric assumption that the lifetime, which is the number of rounds with partially known one-time-pad needed to find f, is linear in the key length. Although our initial experimental lifetime results did not confirm our estimate, we experimentally identified the reason for it and showed that Eve can succeed in forging a message within our lifetime bound.

Therefore, when using this authentication in the context of QKD, the secret hash function must be changed regularly, taking into account that the amount of information allowed to leak in each QKD round is the quantity that controls the lifetime of the system. But, we recommend using the original Wegman-Carter authentication with ϵ-ASU_2 hash functions in combination with the remedy proposed in [3].

5

Concluding Remarks and Future Research

The goals of this thesis were the following. One, to investigate properties of a Strongly Universal$_2$ hash function family to help understand the properties of (classical) authentication used in Quantum Cryptography. Two, to study vulnerabilities of a recently proposed authentication protocol intended to rule out a "man-in-the-middle" attack on Quantum Cryptography [19]. Three, to estimate the lifetime of authentication using encrypted tags when the encryption key is partially known.

For the first goal, we have studied the properties of the family $\mathcal{H}_3$. We have identified Eve's possibilities when her partial knowledge of the secret key satisfies certain conditions (for example, (3.9)). If the conditions are satisfied, then there are messages for which she can generate the correct tag. But it is still difficult for Eve when the family $\mathcal{H}_3$ is used. We also observed an interesting property of each individual hash function in the family $\mathcal{H}_1$. That is, when this family is constructed for special parameters, the uniform distribution on $\mathcal{M}$ induces a $1/|\mathcal{M}|$-almost uniform distribution on $\mathcal{T}$.

For the second goal, we showed that the recently proposed authentication primitive in [19] is insecure when used in a generic QKD protocol. The main problem is that Eve is not limited to a fixed (random) message, but can in fact choose what message to send, and can check if her chosen message gives the same tag as Alice's message, since the first-step hash function f is publicly known. Using extra shared secret key for an extra authentication in one of the phases probably improves the situation, but it should be stressed that, unlike the Wegman-Carter authentication, the security of the proposed authentication procedure is highly dependent of the context in which the authentication is applied.

For the last goal, we estimated the lifetime of the authentication system in question. We showed that the lifetime depends linearly on the authentication key length. The theoretical estimate is supported by experimental results on the family $\mathcal{H}_1$. Although our experimental results were not as predicted by theoretical results under simplifying assumptions, we further presented empirical results to figure out the reasons for the unexpected outcomes. As a consequence, this authentication primitive is not unconditionally

secure if used in an environment where the encryption key (which is the one-time-pad in our case) is partially known in each round. Information leakage in QKD is unavoidable. Eve may have partial information on the generated QKD keys, a portion of which is used as the one-time-pad in each authentication. Therefore, when using this authentication in the context of QKD, the secret hash function must be changed regularly, taking into account that the amount of information allowed to leak in each QKD round is the quantity that controls the lifetime of the system. But, we recommend using the original Wegman-Carter authentication with ϵ-ASU$_2$ hash functions in combination with the remedy proposed in [3].

As an immediate continuation of this research, it would be interesting to study the following. One, to study specific classes of ϵ-ASU$_2$ hash function families that are resistant to the attacks of [3] when used for authentication in QKD. A recently proposed class of ϵ-ASU$_2$ hash functions is Variationally Universal (VU) hash functions, which are stronger than ϵ-ASU$_2$ hash functions [25]. An interesting question would be whether using VU hash functions strengthen the security of authentication in QKD against the attacks of [3]. Two, to formally prove that the protocol of [19] is secure when augmented with additional protocol parts intended for other purposes that have authenticating properties. In addition, to estimate the increase in key consumption rate of such a system. A thorough understanding of the full QKD system is required to achieve this goal. Three, to sharpen the results on authentication using encrypted tags. This would include finding out the actual distribution of the random variable $X_i|X_{i-1} = k$, and using it in the lifetime estimate; identifying the case when Eve is able to generate the correct tag for her message in the lifetime estimate; and also separating the case of small k in the estimate.

To conclude, authentication with reduced key consumption rate may have security vulnerabilities, when used in an environment where information leakage is unavoidable. There are countermeasures that strengthen the security of these types of authentication. For instance, in the case of the recently proposed authentication primitive one countermeasure is to use extra key for an extra authentication purpose. Another example, in the case of authentication using encrypted tags the countermeasures are, among others, reducing the amount of information leakage, changing the secret hash function frequently and so on. Of course, using the original Wegman-Carter authentication with the modification proposed in [3] would restore the security. Nevertheless, it will be interesting to do further research into efficient, less key-consuming authentication with strong security.

Bibliography

[1] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proceedings of the 35th Symposium on Foundations of Computer Science,* Los Alamitos, edited by Shafi Goldwasser, IEEE Computer Science Press, 1994, pp. 124-134.

[2] C. H. Bennett and G. Brassard, "Quantum Cryptography: Public Key Distribution and Coin Tossing," in Proc. IEEE Int. Conf. Comput. Syst. Signal Process., Bangalore , India, 1984, pp. 175-179.

[3] J. Cederlöf, J-Å. Larsson, "Security Aspects of the Authentication Used in Quantum Cryptography," IEEE transactions on Information Theory, Vol 54(2008): s. 1735 - 1741.

[4] J. Cederlöf, "Authentication in quantum key growing," *Master Thesis in Applied Mathematics, Linköping University,* LiTH - MAT - EX - - 05 / 18 - - SE.

[5] N. Gisin, G. Ribordy, W. Tittel and H. Zbindin, "Quantum Cryptography," *Rev. Mod. Phys.,* vol. 74, 2002, pp. 145-195.

[6] J. L. Carter and M. N. Wegman , "Universal Classes of Hash Functions," J. Comput. Syst. Sci., vol. 18, pp. 143-154, 1979.

[7] M. N. Wegman and J. L. Carter, "New Hash Functions and Their Use in Authentication and Set Equality," J. Comput. Syst. Sci., vol. 22, pp. 265-279, 1981

[8] D. R. Stinson, “Universal hashing and authentication codes,” in *Advances in Cryptology: Proceedings of Crypto 91*, J. Feigenbaum, Ed., vol. 576. Springer, 1991, pp. 74-85.

[9] D. R. Stinson, “Cryptography: Theory and Practice,” (Discrete Mathematics and Its Applications).

[10] D. R. Stinson, “Universal hash families and the leftover hash lemma, and applications to cryptography and computing,” in *Journal of Combinatorial Mathematics and Combinatorial Computing*, vol. 42, 2002, pp. 3-31.

[11] D. R. Stinson, “On the connections between universal hashing, combinatorial designs and error-correcting codes,” in *Congressus Numerantium*, vol. 114, 1996, pp. 7-27.

[12] M. Atici and D. R. Stinson, “Universal hashing and multiple authentication,” *Lecture Notes In Computer Science,* Springer-Verlag London, UK, vol. 1109, 1996, pp. 16-30.

[13] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail, and J. Smolin, Experimental quantum cryptography *J. Cryptol.*, Vol. 5, no. 1, pp. 3-28, (1992).

[14] G. Brassard and L. Salvail, Secret-key reconciliation by public discussion, Advances in Cryptology-Eurocrypt ’93, edited by T. Helleseth, *Lecture Notes in Computer Science* Vol. 765 pp. 410-423., (Springer, Berlin, 1994).

[15] C. H. Bennett, G. Brassard and J. M. Robert, Privacy amplification by public discussion, *SIAM J. Comput.*, Vol. 17, no. 2, pp. 210-229., (1988).

[16] C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Mauer, Generalized privacy amplification, *IEEE Trans. on Inf. Theory*, Vol. 41, no. 6, pp. 1915-1923., (1995).

[17] G. Gilbert and M. Hamrick, "Practical Quantum Cryptography: a Comprehensive Analysis", quant-ph/0009027.

[18] H. Barnum *et al*., "Authentication of Quantum Messages" *Proc. 43rd Annual IEEE Symposium on the Foundations of Computer Science (FOCS ’02)*, pp. 449-458. IEEE Press, 2002.

[19] M. Peev *et al*., A novel protocol-authentication algorithm ruling out a man-in-the-middle attack in quantum cryptography, *Int. J. Quant. Inform.* **3**, 225, (2005), quant-ph/0407131.

[20] T. Beth, J. Müller-Quade, and R. Steinwandt, Cryptanalysis of a practical quantum key distribution with polarization-entangled photons, *Quantum Information and Computation* **5**:181-186 (2005), quant-ph/0407130.

[21] M. A. Nielsen and I. L. Chuang, Quantum Computation and Quantum Information, *Cambridge University Press*, (2000).

[22] X. Wang, Y. L. Yin and H. Yu, Finding Collisions in Full SHA−1, in *Advances in Cryptology- CRYPTO 2005,* 25th Annual International Cryptology Conference, Santa Barbara, California, USA, Proceedings, Vol. 3621 of LNCS, (Springer, 2005), pp. 17-36.

[23] C. D. Cannière and C. Rechberger, Finding SHA−1 Characteristics: General Results and Applications, *Lecture Notes in Comp. Sci.*, Vol. 4284, (Springer, 2006), pp. 1-20.

[24] N. Lütkenhaus, Estimates for practical quantum cryptography, *Phys. Rev. A* Vol. 59, (1999), pp. 3301-3319.

[25] T. Krovetz and P. Rogaway, Variationally universal hashing, *Information Processing Letters 100* (2006), pp. 36-39.